# CrowdSim: A Generative Model of Crowdsourced Survey Responses

Michael Lepori, Derek Thayer, Sean Guarino, Leonard Eusebi
Charles River Analytics
Cambridge, MA
mlepori@cra.com, dthayer@cra.com, sguarino@cra.com, leusebi@cra.com

## ABSTRACT

Career development relies on an understanding of possible future roles, available training experiences, and current skills. To target training where it will be most effective, trainees and instructors must understand how these elements interact, which requires analysis of historical data. These data can be difficult and often sensitive to collect from human subjects, requiring longitudinal studies with privacy safeguards. Synthetic data generation that simulates the knowledge and biases of individuals offers an opportunity to develop the algorithms that are most likely to succeed, prior to human-use testing.

This paper presents a generative model of crowd knowledge and responses to bootstrap the evaluation of algorithms whose recommendations or measurements will produce downstream effects. The simulation models the effect of well-known cognitive biases that influence knowledge recall or learning outcomes. For example, the availability heuristic (Tversky and Kahneman, 1973) may cause the importance of an infrequent or highly variable task to be less accurately recalled. Simulations must model the strength and prevalence of each cognitive bias, parameters which may vary with domain and are therefore difficult to set a priori. By adjusting the relative influences of declarative knowledge, anecdotal experience, cognitive biases, and other factors, the generative model creates many simulations with different underlying assumptions, allowing it to probe the generality of algorithms trained or selected using a subset of the simulations. We present an application of this generative model to crowdsourced survey responses. Our approach enables researchers to select the best performing algorithms under a range of psychologically plausible assumptions. Furthermore, the generative model provides an open and scalable source of data that is not constrained by issues of data privacy, security, or collection cost. This model selection approach extends to related domains where relevant psychological effects can be approximated, such as tailoring training recommendations and intelligent tutoring to individual backgrounds.

## ABOUT THE AUTHORS

**Michael Lepori** is a Scientist at Charles River Analytics. Mr. Lepori's primary research interests include machine learning, natural language processing, and cognitive psychology. Mr. Lepori currently works on a project involving crowdsourced job analysis, as well as projects in natural language processing and AI game-playing. His current and past projects have studied the use of synthetic data for interpreting and enhancing black-box models, human perception of machine-generated adversarial stimuli, as well as novelty-aware AI. Mr. Lepori received a bachelor's degree in physics from Johns Hopkins University.

**Derek Thayer** is a Software Engineer at Charles River Analytics. Mr. Thayer's research interests are in machine learning, including deep reinforcement learning, neural networks, and autonomous agents. At Charles River, Mr. Thayer has used deep neural networks to assist in the classification and prediction of survey responses for quality and veracity. Mr. Thayer received a B.Sc in Biophysics from Loyola University Chicago, and a M.Sc in Computer Science from the Georgia Institute of Technology.

**Sean Guarino** is a Principal Scientist and Director at Charles River Analytics. Mr. Guarino's research interests include game design, incentive engineering, intelligent tutoring, and cognitive systems engineering. He oversees projects in each of these domains, including commercial intelligent tutoring and training simulation products. Mr. Guarino holds a B.S. in Computer Science from Cornell University and an A.L.M. in Psychology from Harvard Extension School.

**Leonard Eusebi** is a Senior Scientist at Charles River Analytics. Mr. Eusebi's primary research interests include synthetic data generation, crowdsourcing, game theory, game design, and collective intelligence. Mr. Eusebi currently leads projects in crowdsourced job analysis, applied game theory, and gamified image analysis. Prior to joining Charles River, Mr. Eusebi received bachelor and master's degrees in physics from Wesleyan University, and then spent 8 years as a game designer in the commercial games industry.

# CrowdSim: A Generative Model of Crowdsourced Survey Responses

Michael Lepori, Derek Thayer, Sean Guarino, Leonard Eusebi
Charles River Analytics
Cambridge, MA
mlepori@cra.com, dthayer@cra.com, sguarino@cra.com, leusebi@cra.com

## INTRODUCTION

This paper presents our generative model of survey responses, called the crowd simulator. The crowd simulator can be used to develop an active learning approach to survey distribution by refining the active learning algorithm prior to data collection. The simulation approach models systematic effects that influence survey responses, such as cognitive biases. For example, the availability heuristic (Tversky & Kahneman, 1974) may cause the importance of an infrequent or highly variable task to be less accurately reported. Any one simulation of survey responses must make assumptions about the strength and prevalence of each cognitive bias, parameters which may vary with domain and are therefore difficult to set a priori. By adjusting the relative influences of declarative knowledge, anecdotal experience, cognitive biases, and other factors, the generative model creates many simulations with different underlying assumptions, allowing it to probe the generality of the selected survey distribution algorithm. The generative model provides an open and scalable source of survey response data that is not constrained by issues of data privacy, security, or collection cost.

### Motivation

Surveys are rich sources of human-subjects data, and are regularly employed in a variety of disciplines, such as political science, market research, economics, psychology, and healthcare (Abuse, 2016; Boril et al., 2018; Buskirk & Andrus, 2012; Couper et al., 2001). Typically, a single survey is developed and distributed to a large number of participants, who are all expected to respond attentively to all questions asked. Survey responses are then analyzed in order to generate a composite picture of the "true" answer to all of the questions, perhaps disaggregated by demographic or other factors. In some survey design settings, there might be many questions that we are seeking answers to, resulting in large surveys that impose a burden on individual participants. This burden reduces the ability to conduct surveys, by increasing the cost and by turning away some participants who might complete a shorter survey.

We reframe the problem of survey design to reduce the individual burden on participants. The critical insight is that survey-based data collection is often treated as a batch learning problem, but it is fundamentally an online learning problem. In other words, researchers typically only design a single large survey and distribute this static survey to all participants. However, one can design different (and smaller) surveys targeting different information, and distribute these surveys in small batches, building up better and better estimates for the "true" answer to a question. An adaptive, incremental learning approach to survey design and distribution provides an opportunity to reduce the burden on individual participants and maximize the value of each question asked. Now that we have reframed the problem as an online learning problem, we can further refine the framing by considering the *active learning* setting. In active learning problem settings, the machine learning algorithm receives data in batches in real-time (as in online learning), but the machine learning algorithm is equipped with a means of querying specific information in those batches (a *query algorithm*). In this work, we present an adaptive survey design algorithm using active learning. Specifically, our approach consists of a probabilistic information aggregation algorithm that learns to represent the information that we receive from survey responses, along with a query algorithm that can generate new surveys and collect new information from the crowd.

A critical problem with this setting is that of validating our query algorithm, as the system's efficacy is fundamentally limited by the data that it queries. The typical approach to validation would be to collect data from the real world using a variety of query algorithms, and select the best algorithm, given a specific notion of "best". However, this requires us to define a notion of algorithm quality, which may be difficult without a ground truth distribution, and then collect data from users, which can be costly when selecting between many algorithms. We are faced with a question: Without knowing what the data will be, or which model states you will reach, how do you ensure the appropriate choice of collection algorithm? In this work, we attempt to address that question using a generative model of crowdsourced responses, which is flexible enough to coarsely model response biases and other structured noise that we may anticipate in real-world deployment.

**Novel Contributions**

The present work makes several novel contributions to various fields. First, it makes a theoretical contribution to the survey design literature, reframing the task of distributing surveys as an active learning task. Furthermore, it presents an approach to validating the query algorithm *without ground-truth data*, using a task setting that we call adversarial active learning. Adversarial active learning attempts to probe the generality and robustness of a query algorithm, ensuring that the algorithm is performant under a variety of conditions and thus providing evidence that it will perform well during deployment. We define the adversarial active learning approach as follows: we are given a set $O$ of oracles, which represent different survey populations, a set $Q$ of query algorithms, which design surveys to be distributed to participants, and a set of active learning algorithms $A = (L, q)$ consisting of a fixed learning algorithm $L$ and $q \in Q$. For each $o \in O$ we test the performance of every $a \in A$ and select the best performing active learning algorithm $a$. If the best performing algorithm is stable against a set of oracles, this can increase our confidence in the generality of the algorithm. This form of validation assumes that the oracles are sufficiently diverse, perhaps representing demographic groups or different domains. In the present work, we introduce a crowd simulator that can generate data for different oracles. Though adversarial active learning could, in principle, be performed with oracles that are composed of actual survey participants, using a simulator gives us access to a much larger set of possible oracles, without incurring the costs associated with real-world data collection and without changing our approach to algorithm selection.

The crowd simulator is a general framework for simulating survey responses. The key innovation is that the crowd simulator can model arbitrary correlations between questions or biases when answering a single question. We refer to these effects as systematic effects. Beyond differences between different subpopulations of participants (which correspond to different true distributions over survey responses), these systematic effects can coarsely model a whole range of cognitive or response biases that might impact a participant's response to a question. By parameterizing the crowd simulator such that it can capture a variety of strengths of these systematic effects, we hope to capture and test against data representing much of the variance that we might see in deployment. Though we might only have a coarse notion of how a given systematic effect will impact survey responses (e.g., this effect will cause participants to overestimate the answer to a given question), we can build that systematic effect into the crowd simulator, generate oracles that exhibit this effect with various strengths (e.g., different amounts of overestimation), and then test our active learning algorithms against those oracles. If one active learning algorithm outperforms the rest, then we can be confident that the algorithm will be robust to real-world effects that generally obey this mechanism (e.g., effects that result in overestimation).

**Case Study - Crowdsourced Job Analysis**

We demonstrate our adversarial active learning task and survey response simulator in the domain of crowdsourced job analysis. Job analysis is a formal procedure for defining a job in terms of the tasks that compose it, as well as the qualifications that it requires. Job analysis is the basis of many human resources functions, including candidate selection, promotion, training and development, compensation, designing rewards programs, and performance management. Aside from its critical role in human resources activities, job analysis is legally required by the Federal Government (Thompson & Thompson, 1982; Warren, 1976).

Traditionally, job analysis is performed by expert job analysts, which requires great time and effort on their part, and comes at a steep cost for organizations. Furthermore, job analysts typically conduct interviews and use a series of long survey forms to perform their analyses, which prevents regular updates to job analysis. We are developing a crowdsourcing approach to this analysis, which can be conducted by non-experts and minimizes the disruption to employees by distributing the effort across a larger group. This approach allows the analysis to be updated regularly, preventing out-of-date information from skewing the results. Eliminating stale data on required tasks, skills, and contextual factors of a job can ensure that assignments are accurate and training curricula are kept up to date.

In the present work, we focus on refining estimates of two essential task characteristics: task frequency and task importance. For our purposes, we assume that the list of relevant tasks has already been established. Furthermore, we assume that our data exhibits the following constraints: 1) the data is collected sequentially, 2) frequency and importance survey questions exist on a Likert scale, 3) each participant has time for a fixed number of questions, 4) the survey distribution system may process the data from one individual before generating the next survey, 5) there are a limited number of individuals available to query.

## BACKGROUND

### Crowd Simulation – Modeling Systematic Effects

Many studies have shown the existence of systematic effects – correlations between questions or systematic biases – that can influence question responses in predictable ways (Burkner, 2019; de Ayala, 2009; Friesen & Weller, 2006). Though we may not know the magnitude of the effect a priori, we may hypothesize about the context in which the effect happens and the directionality of the effect using qualitative findings from the literature. By including these biases in our simulated survey responses, we can generate more realistic adversarial tests of our active learning algorithms. In the present work, we demonstrate systematic effects that loosely mimic the effects of two well-known cognitive biases – the availability heuristic and the anchoring and adjusting heuristic.

The availability heuristic is a well-known and long-studied cognitive bias that states the perceptions of an event are impacted by ease in which a subject can recall instances of that event. One experimentally-verified consequence of this bias is that events that are more easily recalled are thought to occur more frequently (Tversky & Kahneman, 1973). This effect has been studied across a variety of different domains, including media, economics, and criminal justice. We model this in our simulator by increasing the perceptions of task frequency if a participant can easily recall instances of performing that task. The probability of this recall is increased for higher-frequency tasks.

The anchoring and adjusting bias is another well-known cognitive bias that states that a judgement is influenced by a previous judgement, which serves as an anchor (Block & Harper, 1991; Epley & Gilovich, 2006). One common mechanistic explanation of this effect is that the anchor serves as a reference point for the next judgement. When making that second judgement, a participant will refer to their anchor and then adjust it to produce the second judgement. Oftentimes, the participant fails to sufficiently adjust, which biases responses in the direction of their previous judgement. This effect is known to be moderated by the similarity of the questions requiring judgements (Mussweiler, 2002). In our context, this means that the answers to successive questions will be impacted by their answer to the previous question. This will be moderated by the similarities between the questions.

### Survey Generation - Active Learning

The adversarial active learning paradigm that we introduce is a type of active learning (Settles, 2009), which itself is a type of online learning (Diethe & Girolami, 2013; Jain et al., 2014). Online learning is a paradigm in machine learning where the machine learning algorithm is exposed to small, sequential batches of data due to some constraints on the data pipeline. Online learning algorithms must be able to continuously learn and produce predictions. Active learning is a paradigm that equips an online learning algorithm with a query algorithm, which enables the system to query an oracle for data that is most useful for the machine learning algorithm. Thus, an online learning algorithm needs to process data in sequential batches without any influence over the contents of those batches, whereas active learning algorithms can query for specific batches.

In the context of crowdsourced job analysis, we treat whole populations of incumbents as the oracle for learning a definition of a job. Surveys can only be distributed in small numbers to portions of this crowd, which results in sequential, batched data. To reduce the burden of surveys on individuals, we seek to reduce survey size by actively selecting the most informative survey questions at each batch, rather than pre-defining surveys. We then continuously train a probabilistic machine learning algorithm based on these survey response batches. This setting maps directly onto the active learning paradigm. Our survey generation algorithm is our query algorithm, our probabilistic algorithm is our machine learning algorithm, the subpopulation corresponds to our oracle, and each batch of survey responses corresponds to our sequential batch of data.

## METHODS

We now formally define the various components of our adversarial active learning system as applied to crowdsourced job analysis. This includes our parameterized crowd response simulator, our parameterized survey generator, which is comprised of a probabilistic machine learning algorithm (Job Definition Model) and a parameterized query algorithm. We will use these components to measure the relative performance of different survey generators (which each have different query algorithms) across different crowd simulator parameterizations (which have different systematic effects). Using this approach, we can assess the effectiveness of different survey generators on a variety of potential subpopulations, which may exhibit different systematic effects, and may also have different distributions of true answers for each survey question. This approach tests our assumptions about how different query algorithms will improve active learning without first performing costly data collection. An algorithm that performs as expected across a variety of crowd simulator tests can be deployed with some confidence that it will perform well.

**Crowd Simulator**
The crowd simulator is a general framework for simulating survey response data. This framework can generate sample answers to any Likert or true/false question. Crucially, the crowd simulator framework can also model correlations between question responses or response biases that may impact individual question responses. For brevity, both correlations and response biases will be termed *systematic effects*. Each systematic effect provides a series of hyperparameters, allowing users to explore different configurations of magnitude and prevalence for every systematic effect. By creating multiple crowd simulator instances with different hyperparameters, one can simulate collecting data from different possible subpopulations, which each exhibit different systematic effect strength and prevalence. We formally define the crowd simulator framework below.

Define a survey **S** as an ordered collection of **N** survey questions $\mathbf{Q_i}$, $i \in [1, N]$. For each $\mathbf{Q_i}$, define a continuous random variable $\mathbf{T_i}$, which represents a user's *true answer* to that question. Define $\mathbf{T_i}$'s PDF as $P(T_i)$. Thus, we assume that there is no one correct answer for a given question, but that different users will experience different true answers. The support of each $\mathbf{T_i}$ may take on any range of values, if the possible Likert responses correspond to values in that range. The range typically depends on the content of question $\mathbf{Q_i}$. For example, if $\mathbf{Q_i}$ asked how frequently a task was done, then the support might range from .5 (the task is completed twice a day, the highest Likert value) to 365 (the task is completed once a year, the lowest Likert value). If $\mathbf{Q_i}$ asked how important a task is, then the support might just be a continuous distribution over the possible Likert values (ex. [0.5, 5.5] for a 1-5 Likert scale).

For each user (indexed by j), draw $t_{ij} \sim T_i$, and define $N(t_{ij}, \sigma_i)$, which represents the *response distribution* for a given user and question. This distribution encodes any uncertainty that the user might have in reporting their true answer on a survey. Next, we define **K** pairs of systematic effect functions: $f_{tk}(t_{ij}, \sigma_i, \theta, S) = t_j + \Delta_{tk}$, $f_{\sigma k}(t_{ij}, \sigma_i, \theta, S) = \sigma_i + \Delta_{\sigma k}$. Each $f$ takes in the mean and standard deviation of the response distribution for the current question and user, a set of function specific hyperparameters θ, and information about other questions in the same survey **S** (including whether they have been answered, and what those answers were), and adds a (not necessarily positive) value to the mean or standard deviation, respectively. Collapsing the effects of all K pairs of systematic effect functions, we specify the *effective response distribution*: $N(t_{ij} + \Delta_t, \sigma_i + \Delta_\sigma)$. Finally, we draw the user's response $r_{ij} \sim N(t_{ij} + \Delta_t, \sigma_i + \Delta_\sigma)$. We apply a rounding function $ROUND(R_{ij})$ to generate the Likert response for $\mathbf{Q_i}$, user j. In order to generate responses for a full survey, we iterate through this process in order, generating responses for $\mathbf{Q_1}$ through $\mathbf{Q_N}$. Generating these responses in order is critical, as the information **S** passed to the systematic effects functions contain information about all survey responses for questions $\mathbf{Q_1}$ through $\mathbf{Q_{i-1}}$.

Now, we define the systematic effect functions that loosely mimic the availability heuristic and anchoring and adjusting heuristic. We emphasize that we are not making any claims to provide an experimentally verified model of the effect of these heuristics, nor are we making any concrete mechanistic claims about them. We only seek to use these well-known cognitive biases as inspiration to demonstrate the capabilities of the crowd simulator, and to demonstrate the utility of the adversarial active learning paradigm.

First, we consider the availability heuristic. Intuitively we want to model the scenario where a participant has an easily recallable instance of performing a task, which biases their frequency response in a positive direction. Let $\theta = \{N, S_{AH}\}, S = \{\}$. $N$ is an integer parameter encoding how many days in the past a participant considers to be "recent". Thus, if $N = 5$, then any task done within the past 5 days is considered recently done. $S_{AH}$ is a parameter encoding the strength of the availability heuristic. We then define:

$$p_{rd} = N/t$$

Where *t* corresponds to the true period of a given task, in days, for a given participant. Thus, if the task is performed once a work week, then $t = 5$. $p_{rd}$ represents the probability that a task was "recently done". We clamp the range of this value to [0, 1]. We then draw:

$$rd \sim Bernoulli(p_{rd})$$

Which defines whether the task was easily recalled. If rd = 1, then $f_{tk}(t_{ij}, \sigma_i, \theta, S) = t_j + S_{AH}$, else $f_{tk}(t_{ij}, \sigma_i, \theta, S) = t_j$. For completeness, we note that $f_{\sigma k}(t_{ij}, \sigma_i, \theta, S) = \sigma_i$.

Next, we consider the anchoring and adjusting heuristic. We have already noted that the similarity of judgements is a moderator in the effect of the anchoring and adjusting heuristic. For this study, we encode this in our function by making the strong assumption (which is likely invalid in real populations) that the anchoring and adjusting heuristic

only applies when subsequent questions are of the same type (e.g., both frequency, both importance, or both day one importance). Let $\theta = \{S_F, S_I, S_{D1}\}, S = \{R^{-1}\}$. The S values determine the strength of the anchoring and adjusting effect, which might vary between question types. The R value encodes the question type and response to the previous question. Let type() be a function that returns the question type for a parameter. The function $f_{tk}(t_{ij}, \sigma_i, \theta, S)$ is then defined as follows:
IF $(type(t_{ij}) = type(R^{-1})$: return $t_{ij} - R^{-1} * S_{Type(t_{ij})}$
ELSE: return $t_{ij}$
Thus, the function biases the response of a subsequent question to be closer to the response of a previous answer if both questions are of the same type and has no effect if they are of different types. For completeness, we note that $f_{\sigma k}(t_{ij}, \sigma_i, \theta, S) = \sigma_i$.

**Survey Generator**
We formalize the survey generation system to be a function from the space of task characteristics to a space of queries. In our case, characteristics are the current estimates of frequency and importance from our probabilistic machine learning algorithm (job definition model), and the queries are the survey design. Each survey generator consists of a job definition model and a unique query algorithm, which selects a survey design to be distributed. The query algorithm defines a cost function, which is then used to rank potential survey configurations.

The job definition model (JDM) is updated by active learning data collection. Specifically, the JDM is a Bayesian network, where the probability distributions for each question are defined over the Likert scale of possible responses for the question. The JDM counts occurrences of question responses and builds up a probability distribution representing the probability that question's true answer. For each possible survey question, the JDM calculates an estimate of the value of asking that question in the next batch of surveys. The value calculation is simply the inverse Kullback-Leibler (KL) Divergence between the JDM's current probability distribution and a discrete uniform distribution defined over the same Likert scale. Intuitively, this encodes the amount of information that the JDM has about a survey question. When the JDM is confident in a question's answer, then the KL divergence between that distribution and the uniform will be high.

Each question is assigned a baseline cost by the JDM equal to the current KL Divergence. This cost is then modulated by a series of parameters that define a query algorithm's cost function. Each survey questions considers only one of the characteristics of a task: the frequency or the importance. For each possible characteristic, we define a cost parameter that scales the baseline value of any survey question querying that characteristic. This allows the survey generator to emphasize one characteristic over the other or to balance between them. We also define pairwise cost parameters that consider whether the previous question asked about the same characteristic or a different one. This allows the survey generator to emphasize switching or not switching between characteristics with each query.

The cost function operates on a whole survey, considering the ordering of each questions as well as the adjusted individual costs of those questions. The survey generator optimizes this process by first sorting on the baseline cost, randomizing the order of questions with the same baseline cost. Then the list of questions is divided into separate lists for each characteristic and the survey generator calculates the cost of all surveys that can be built by choosing the next element from one of these lists. Since we are concerned with distributing short surveys, a full enumeration of these possible survey orderings is possible. This defines the process for generating a survey, given probabilistic estimates of question values from the JDM and a parameterized cost function.

**Experimental Setup**
We now define how these components combine in our experimental setting. First, we define the number of tasks that our system will attempt to learn about, as well as the mean of their true answer distributions. Unless otherwise noted, our surveys ask about 20 tasks, for a total of 40 questions (one frequency and one importance question per task). For each question, we use a random number generator to generate the means of the true distribution and use a standard deviation of 0.1 for frequency and importance questions. Next, we define an oracle, which consists of a particular parameterization of crowd simulator. This involves selecting the parameters for each systematic effect (availability heuristic and anchoring and adjusting heuristic), as well as the standard deviation parameters that define the response uncertainty. Defining these parameters provides us with an oracle that represents a particular subpopulation that we may encounter. Finally, we define the cost function parameters space for the query algorithm of the survey generator. These parameters define the constants that determine how new surveys are generated.

For each run, the crowd simulator and survey generator cost parameters are held constant, but a different task model is used, assigning different values to the true mean of each characteristic of each task. The set of task models used is held constant across different parameterizations of the crowd simulator or survey generator, to allow clearer comparison. The experimental loop proceeds as follows: First, the survey generator creates a survey, which is realized by a set of parameters passed to the crowd simulator. The crowd simulator generates responses to every question in the survey. Then, the Job Definition Model learns from these responses, produces estimates of the value of asking each question for each task, and passes these estimates on to the survey generator query algorithm. The query algorithm's cost function then determines the best ordering of questions for the next user. Finally, the survey generator creates a new survey, and the loop restarts. This loop continues until a fixed number of surveys have been distributed.

We perform runs with 60 users, 5 task models, 20 tasks, and a survey length of 5. For each baseline comparison run, we use a survey length of 40, allowing all questions to be asked of all users. The crowd simulator parameterizations are "No Effects", "AH", "AA", and "All Effects". In the "No Effects" parameterization, users report noisy estimates of the true mean value for each task characteristic query. In the "AH" parameterization, users' responses are affected only by the availability heuristic effect described above. In the "AA" parameterization, the users' responses are affected only by the anchoring and adjusting effect, with anchoring occurring only for consecutive questions about the characteristic (e.g., a frequency question followed by another frequency question). In the "All Effects" parameterization, user responses are affected by both the availability heuristic and the anchoring-and-adjusting effects.

The survey generator parameterizations are "Random", "Base", "Freq", "Diff", and "All". The "Random" parameterization sets the baseline cost for all task characteristic questions to zero, selecting a survey ordering entirely at random but still compiling results using the JDM. Random sampling is a gold standard approach to learning population means and should perform well given enough users. The "Base" parameterization sets all cost function parameters to the identity, choosing a survey ordering that contains the individual questions with the lowest cost. This approach should allow smaller numbers of users to produce a better estimate across many tasks. The "Freq" parameterization functions like the "Base" parameterization but reduces the estimated cost of all frequency questions by a factor of 4. This should provide more data on the frequency of a task, compensating for uncertainty due to recency bias or the availability heuristic. The "Diff" parameterization function like the "Base" parameterization but increases the cost of a question by a factor of 4 if it is immediately preceded by another question about the same characteristic. This should tend to produce surveys that alternate characteristics, reducing anchoring and adjusting bias. The "All" parameterization includes the cost adjustments from both the "Freq" and "Diff" parameterizations.

### Evaluation Metrics

Our central thesis is that understanding systematic effects and survey constraints will allow us to craft a query algorithm that improves the estimated values of the Job Definition Model (JDM). We measure performance based on the completeness of the final estimates of each question type for each task and the stability of the query algorithm across different crowd simulator variants. Completeness is measured by averaging the 50th, 75th, and 90th percentile accuracy values across five difference task models. This measure was chosen to aggregate the algorithm's ability to avoid large errors for any tasks and its performance on the median task. Accuracy is defined by the KL divergence between the JDM's estimated probability distribution and the true discretized probability distribution of the underlying task model. In order to evaluate the stability of different survey generators, we compare their rank-order performance as a function of crowd simulator parameter changes. Intuitively, this corresponds to analyzing whether the *relative* performance of different survey generators stays the same when learning from different subpopulations. To scale the performance, we also calculate the accuracy values and corresponding completeness measure for a Baseline estimate compiled by simulating responses from every user to every possible question. This Baseline requires too much effort from each user to be applied in the real world but provides an upper bound on the accuracy of a simple survey.

We test three specific hypotheses that provide insight into the value of the crowd simulator: (1) some survey generators will have measurably more stable performance across different crowd simulator; (2) addressing more of the constraints and systematic effects that are present will yield better performance; and (3) addressing fewer constraints and systematic effects that are not present will yield better performance.

## RESULTS

Figure 1 gives an overview of the performance of three possible survey generators across two different crowd simulators. Results are scaled to the baseline value generated by the "No Effects" crowd simulator and the performance progress is captured after every 10 users. 4 different runs were performed using different sets of randomly generated task models. The performance of these runs was averaged to produce the overview graph. From this graph, we can see that performance of the "Random" survey generator is very bad for low numbers of users and is worse relative to baseline for the "All Effects" crowd simulator than for "No Effects". The "Base" survey generator has significantly better performance for the "No Effects" crowd simulator, but worse performance for the "All Effects" crowd simulator. Finally, we see that the "All" survey generator has better and more stable performance across both crowd simulator variants.

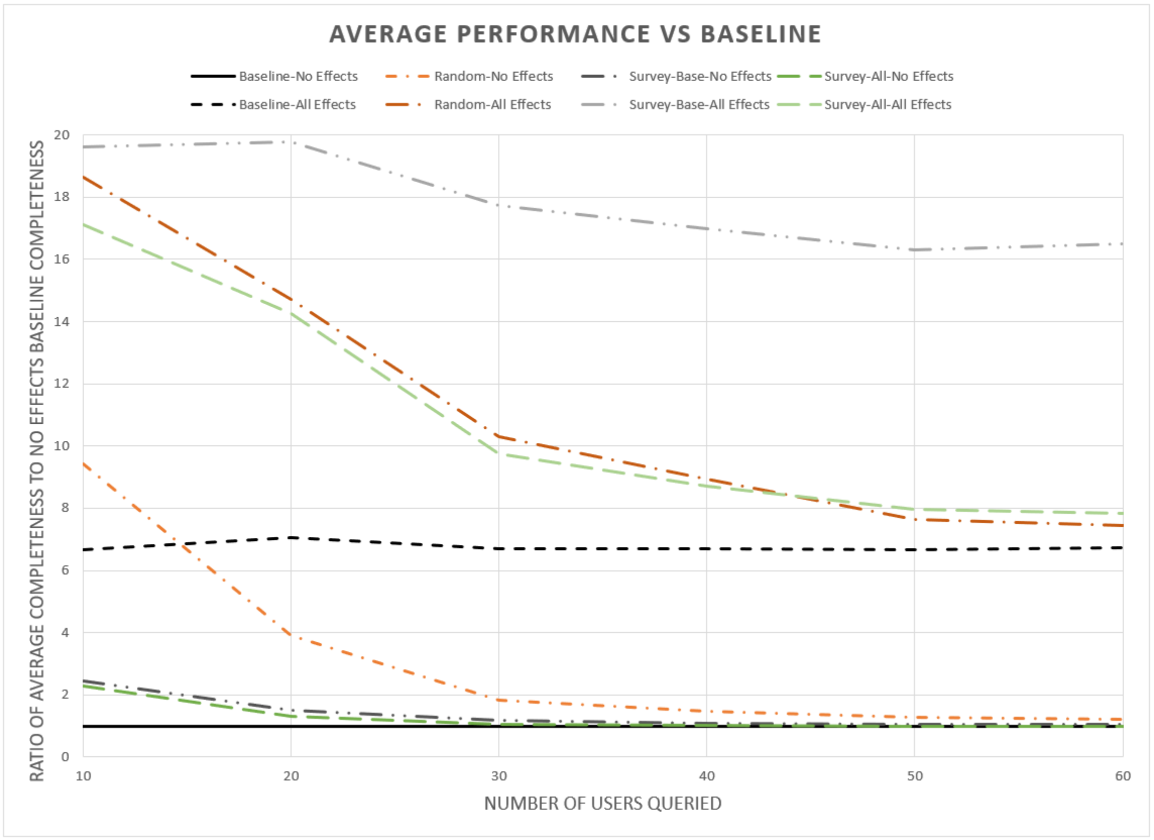

Figure 1: Average Performance vs Baseline Completeness

Figure 2 shows the variability of performance relative to the baseline across different sets of task models for the "All Effects" crowd simulator. This variability helps distinguish meaningful differences between algorithms. For example, the results for the "Random", "Diff", and "All" survey generator parameterizations tend to fall in the same range.

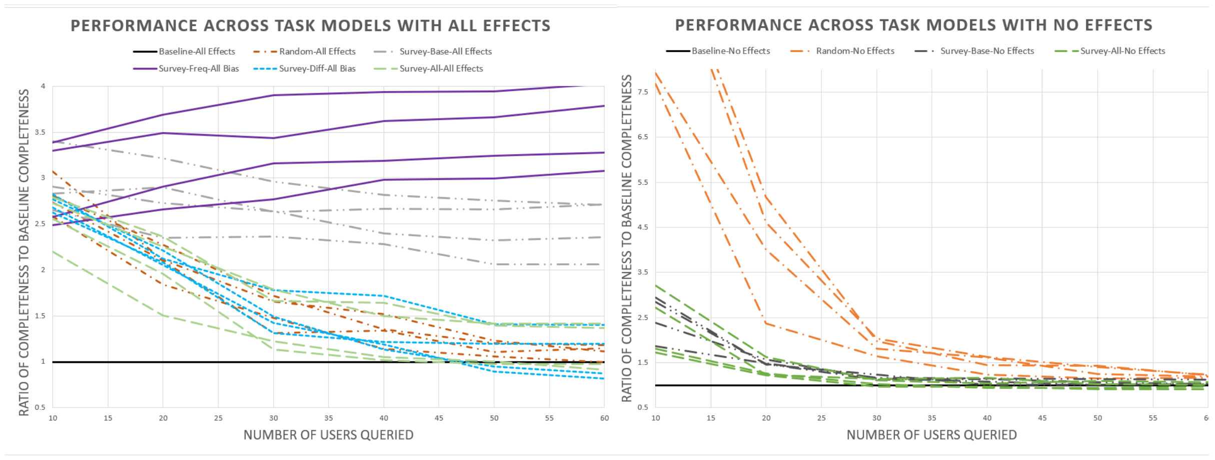

Figure 2: Performance across different task models

Figure 3 shows how performance of the Baseline and the "Random", "Base", and "All" survey generator parameterizations varies across the different crowd simulator parameterizations. Across all data points, the Baseline estimates perform best, as expected, with an average rank order of 1.25 (out of 4). The "All" survey generator has an average rank order of 2.25, "Random" has an average rank order of 3, and the "Base" survey generator has the worst performance, with an average rank order of 3.5.

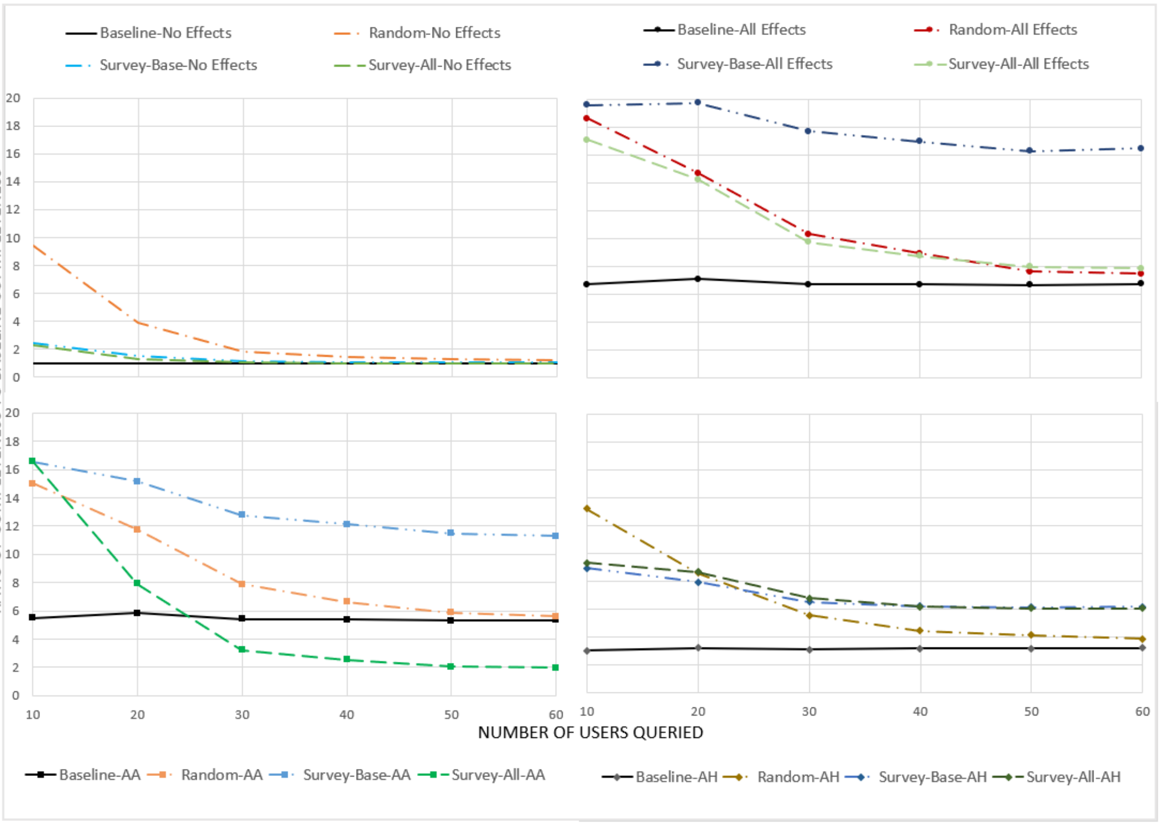

Figure 3: Performance of survey generator parameterizations vs. crowd simulator parameterizations

Based on our hypotheses, we would expect that the "Base" survey generator would outperform "Random" in all of these cases, focusing attention on the questions where users disagree to compensate for the survey size constraint. We would expect the "All" version to outperform "Base" when all effects are present, and to have more varied performance across the different crowd simulator parameterizations (because they will align more- or less-effectively with its cost function). Instead, we see more stable performance from “All” than from “Base” or “Random” and we see that “Random” tends to outperform “Base” whenever systematic effects are present.

## DISCUSSION

### Hypotheses

Our first hypothesis was that some survey generators will have measurably more stable performance across different crowd simulator parameterizations. This hypothesis was shown to be true by the stable rank-order performance of the “All” version of our survey generator. This is a key finding that allows crowd simulator to be used to refine a query algorithm by isolating parameter choices that improve performance across multiple subpopulations.

Our second and third hypotheses were not shown to be true. Addressing more of the constraints and systematic effects that are present does not necessarily yield better performance, since “Random” often out-performed “Base” even though “Base” addressed more constraints in all cases. Likewise, addressing fewer constraints and systematic effects that are not present will not necessarily yield better performance, since the “All” survey generator showed equal performance to the “Base” survey generator in the case where no systematic effects were present. That these two assumptions about how query algorithms should perform proved to be incorrect shows the power of crowd simulator to test our assumptions and help refine our algorithms.

### Implications for Job Analysis

We were able to distill some useful insights about effective query algorithms by using the crowd simulator: (1) when the true distribution of a characteristic is spread across multiple response values, there is an upper bound on the relative entropy of our estimates vs. a uniform distribution, causing those questions to have low costs and therefore to be asked repeatedly; (2) if survey ordering can affect accuracy, it is no longer obvious what the appropriate baseline comparison is, since there is no natural ordering of questions in the same way that there is a natural choice to “ask all questions”. The crowd simulator gives us a method of exploring true accuracy, but a real-world data sample will not.

The present work contributes to the job analysis literature by presenting an active learning formulation of the problem of crowdsourced job analysis. Without a means of validating the query algorithm this formulation would have little practical application. With the introduction of the crowd simulator, we can challenge the robustness of a candidate query algorithm when faced with many different subpopulations, without the need to conduct expensive crowd studies. This allows the cost of job analysis to be reduced to the point that maintaining up-to-date and demographically-debiased job definitions across an entire organization is a feasible goal.

### Generality of Results

The present work advances the machine learning and simulation literature in several ways. First, it introduces a probabilistic generative model of survey responses, which can simulate arbitrary systematic effects, including cognitive biases, correlations between questions, and more. This provides both a theoretical advancement, by connecting the large literature on systematic response effects to the simulation of survey response, and a practical advancement by implementing a simple and extensible framework for simulating these systematic effects. We hope that others use our simplified example systematic effects, which coarsely model two well-known cognitive biases, as inspiration for developing more complete and realistic models of systematic effects, which will increase the real-world applicability of our system. However, even without such development, our framework provides a means of testing the robustness of our survey generator algorithms.

Additionally, we introduce the paradigm of adversarial active learning. Active learning has long been known as a useful strategy for increasing data efficiency in an online learning setting, except its efficacy is limited by the query algorithms that it employs. Choosing the correct query algorithm is thus a crucial part of designing a successful system. The adversarial active learning paradigm is a general method for choosing query algorithms. This procedure uses a parameterized data generation system to simulate datasets with varying characteristics in order to select the most robust query algorithm. Through our experimental results, we have demonstrated that that adversarial active learning provides a cost-effective method for validating certain classes of active learning algorithms prior to expensive or sensitive real-world data collection.

## FUTURE WORK

There are two clear paths to improving the current system: 1) making improvements to the crowd simulator and 2) making improvements to the survey generator. Because we were simply demonstrating the power of a generative probabilistic crowd response simulator, there are many other systematic effects that might impact survey responses. There is a large literature on cognitive biases, and coarsely modeling a wide range of them will certainly improve the real-world robustness of a system such as the one that we presented. In addition, systematic effects such as demographically driven differences in tasking, training, or perceived skill levels could prove fruitful in identify unintended bias produced by a query algorithm (even one as simple as random sampling). The survey generator might be extended to take advantage of knowledge about how skills relate to question responses for different types of tasks. This will be useful especially if the crowd simulator is extended to consider skills. Additionally, one might consider relaxing some of the constraints on the survey generator's survey designs to allow more control over question structures. For example, one might imagine a survey generator that could choose to produce categorical, Likert, or open-ended questions or perhaps even mix between those types for certain characteristics.

## ACKNOWLEDGEMENTS

This work has been supported by Army Research Institute for the Behavioral and Social Sciences contract #W911NF-20-C-0024. The views, opinions, and/or findings contained in this paper are those of the authors and shall not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documents.

## REFERENCES

Abuse, S. (2016). 2015 national survey on drug use and health: Detailed tables.
Block, R. A., & Harper, D. R. (1991). Overconfidence in estimation: Testing the anchoring-and-adjustment hypothesis. Organizational Behavior and Human Decision Processes, 49(2), 188–207.
Boril, J., Smrz, V., Petru, A., Blasch, E., Leuchter, J., Frantis, P., & Jalovecky, R. (2018). Survey of Spatial Disorientation and Sensory Illusion among Air Force Pilots. 2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC), 1–7.
Burkner, P.-C. (2019). Bayesian Item Response Modelling in R with brms and Stan. ArXiv:1905.09501v1, stat.CO.
Buskirk, T. D., & Andrus, C. (2012). Smart surveys for smart phones: Exploring various approaches for conducting online mobile surveys via smartphones. Survey Practice, 5(1), 1–11.
Couper, M. P., Traugott, M. W., & Lamias, M. J. (2001). Web Survey Design and Administration. The Public Opinion Quarterly, 65(2), 230–253.
de Ayala, R. J. (2009). The Theory and Practice of Item Response Theory. The Guilford Press.
Diethe, T., & Girolami, M. (2013). Online learning with (multiple) kernels: A review. Neural Computation, 25(3), 567–625.
Epley, N., & Gilovich, T. (2006). The anchoring-and-adjustment heuristic: Why the adjustments are insufficient. Psychological Science, 17(4), 311–318.
Friesen, G., & Weller, P. A. (2006). Quantifying cognitive biases in analyst earnings forecasts. Journal of Financial Markets, 9(4), 333–365.
Jain, L. C., Seera, M., Lim, C. P., & Balasubramaniam, P. (2014). A review of online learning in supervised neural networks. Neural Computing and Applications, 25(3), 491–509.
Mussweiler, T. (2002). The malleability of anchoring effects. Experimental Psychology, 49(1), 67.
Settles, B. (2009). Active learning literature survey.
Thompson, D. E., & Thompson, T. A. (1982). Court standards for job analysis in test validation. Personnel Psychology, 35(4), 865–874.
Tversky, A., & Kahneman, D. (1973). Availability: A heuristic for judging frequency and probability. Cognitive Psychology, 5(2), 207–232.
Tversky, A., & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. Science, 185(4157), 1124–1131.
Warren, W. H. (1976). " Albemarle v. Moody": Where It All Began. Labor Law Journal, 27(10), 609.