Exploratory Training: When Annotators Learn About Data [*]

Rajesh Shrestha SHRESTHR@OREGONSTATE.EDU
Oregon State University

Omeed Habibelahian HABIBELO@OREGONSTATE.EDU
Oregon State University

Arash Termehchy TERMEHCA@OREGONSTATE.EDU
Oregon State University

Paolo Papotti PAPOTTI@EURECOM.FR
EURECOM,Biot,France

Abstract

ML systems often present examples and solicit labels from users to learn a target model, i.e., active learning. However, due to the complexity of the underlying data, users may not initially have a perfect understanding of the effective model and do not know the accurate labeling. For example, a user who is training a model for detecting noisy or abnormal values may not perfectly know the properties of typical and clean values in the data. Users may improve their knowledge about the data and target model as they observe examples during training. As users gradually learn about the data and model, they may revise their labeling strategies. Current systems assume that users always provide correct labeling with potentially a fixed and small chance of annotation mistakes. Nonetheless, if the trainer revises its belief during training, such mistakes become significant and non-stationarity. Hence, current systems consume incorrect labels and may learn inaccurate models. In this paper, we build theoretical underpinnings and design algorithms to develop systems that collaborate with users to learn the target model accurately and efficiently. At the core of our proposal, a game-theoretic framework models the joint learning of user and system to reach a desirable eventual stable state, where both user and system share the same belief about the target model. We extensively evaluate our system using user studies over various real-world datasets and show that our algorithms lead to accurate results with a smaller number of interactions compared to existing methods.

Keywords: Human-ML Collaboration; Human Learning; Active learning; Human-in-loop ML, Game Theory

[*]. The full version of this paper is published at the proceedings of ACM SIGMOD 2023.

1. Introduction

ML Systems often actively present examples and solicit labels and training examples from users to learn a target model, e.g., *active learning* Aggarwal et al. (2014); Settles (2009, 2012); He et al. (2016); Dimitriadou et al. (2014). In each interaction, the system selects an example for labeling based on the training data already given by the user and the effectiveness of the current model. This may significantly reduce the amount of training data, which is arguably the most expensive resource in learning a model Aggarwal et al. (2014).

Due to the sheer volume and complexity of data, in many settings labeling is often exploratory: users have to explore and learn about the underlying data to label examples correctly. This often happens when the target models are complex or labeling an example requires knowledge about other examples in the dataset. Users learn about the data by repeatedly inspecting it to improve the knowledge and revise the hypotheses about the target model based on the results of preceding interactions. As an example, consider a user who prepares a dataset about patients for some downstream analysis. The goal is to clean the original dataset as this is known to be noisy, i.e., it contains erroneous values. In this task, the user may use an error detection system that needs training data, i.e., labeled examples of clean and noisy tuples, to build its internal model for error detection Mahdavi et al. (2019); Heidari et al. (2019); Thirumuruganathan et al. (2017). After checking a few records, the user starts labeling some values as incorrect as they report abnormally high values for one health indicator. These training samples steer the system to label tuples with high values for such indicator as likely to be mistakes. However, after some interactions, the user may find out that these "incorrect" values are for patients in an age range when those values are actually common. This new information makes some of the labels for the early iterations incorrect, as the user learnt about the data during the annotation process. As another relevant scenario, users may also have to refresh their knowledge about the data and target model due to rapid and frequent data evolution.

As users explore and interact with the data, their knowledge about the data and the target concept evolves. Thus, they may modify their labeling and training strategies. It is known that human learning in interactive settings is highly non-stationary Niv (2009a,b); Young (2004). For example, they may have a prior belief (distribution) about the properties of positive or negative examples for the target concept based on their background information and expertise. They may gradually revise their belief, and in turn labeling and training strategies, after observing new data items, to then settle on a relatively fixed strategy after gaining sufficient knowledge about their actions and environment Niv (2009a).

Current active learning systems often assume that users have perfect knowledge about the target model and underlying data. They usually assume that users provide reliable feedback or training data for target insights with potentially a very small and fixed amount of noise, i.e., stationary user model Aggarwal et al. (2014); Dimitriadou et al. (2014); Golovin et al. (2010); Çetintemel et al. (2013); Li et al. (2012); Zhang and Chaudhuri (2015); Shivaswamy and Joachims (2015). However, as users' knowledge about the data and insight may significantly change during the analysis, it is not clear which inputs are sufficiently reliable for predicting the target models. Due to the significance and non-stationarity of errors, current systems use incorrect labels and learn inaccurate models. Stationary learners are too biased to the initial observations and fail to adapt subsequently. Thus, current methods may learn a model based on early interactions and not use updated knowledge during training.

We argue that, to support users in the best way possible, we should enable them to develop accurate understanding of the target model during training by building learning methods that are aware of how humans learn. In this paper, we set forth novel principles and methods to build systems that enable effective exploratory training. Our key idea is to redesign current learning methods to adapt to users' learning effectively. We leverage the extensive body of work on human learning in psychology and experimental economics to model users' learning during training Niv (2009a); Young (2004). This novel approach views exploratory training as the interactive game with identical interest of two learning agents, i.e., user (as trainer) and system (as learner), for achieving the common goal of learning users' desired models.

2. A Game-Theoretic Framework for Exploratory Training

Agents and Actions. The game of exploratory training, (*ET* for short) has two agents: trainer, i.e., user, and active learner (learner for short). The game is a sequence of interactions at discrete times $t > 0$. Each interaction t starts by the learner presenting a given fixed number k of examples from the dataset to the trainer. The trainer labels these examples according to its current knowledge about the data and the target model. Hence, in each interaction, the *actions of learner* are (to pick) k tuples from the dataset and the *actions of trainer* are (to deliver) k labeled tuples in which each tuple is assigned a label from set $\{0, 1\}$. We assume that the learner provides a fresh example in each interaction. We call each example and its label a *labeling.*

Policies. The *policy* of an agent in each interaction decides which actions the agent performs in that interaction. More precisely, a policy of the trainer in interaction t is a mapping from the set of examples to the set of labels, which assigns a single label to each example. We denote this policy as $\pi_t^{\mathcal{T}}$. The policy of the learner in interaction t, $\pi_t^{\mathcal{L}}$, is a probability distribution over the set of examples, where the probability of an example being in a class indicates the chances that it will be presented to the trainer in interaction t.

Beliefs. Due to their incomplete information about accurate labelings, the agents evaluate the ability of a policy to result in accurate labels based on their current *beliefs* about the target model. The trainer and learner maintain their own beliefs about the model. Let $\mathcal{T}$ and $\mathcal{L}$ denote the trainer and learner agents, respectively. Every agent $i \in \{\mathcal{T}, \mathcal{L}\}$ maintains a *belief* θ_t^i about the target model in interaction t. Given an example, each belief provides a probability distribution over its possible labels, where the probability of each label reflects the agent's confidence in its accuracy. Both agents start the interaction with a *prior belief* θ_0^i. The learner presents and trainer labels examples based on their beliefs.

Payoff of Trainer Policy. The *payoff of each labeling* (x, y) by the trainer in interaction t is the probability by which $\theta_t^{\mathcal{T}}$ deems y to be the accurate label for x, i.e., $\theta_t^{\mathcal{T}}(y \mid x)$. The *payoff of a policy* $\pi_t^{\mathcal{T}}$ in interaction t is the sum of payoffs of its labelings. More precisely, it is $u_T(\theta_t^{\mathcal{T}}, \pi_t^{\mathcal{T}}) = \sum_x \theta_t^{\mathcal{T}}(\pi_t^{\mathcal{T}}(x) \mid x)$ where x goes over all examples shown in interaction t.

Payoff of Learner Policy. The goal of the learner is to form a belief that is in agreement with the trainer's and predicts the labels provided by the trainer accurately. Thus, the closer the learner's belief is to the trainer's labelings in an interaction, the higher the payoff of the learner should be. Let the trainer assign label y to the presented example x in interaction t. This example receives the payoff of $\theta_t^{\mathcal{L}}(y \mid x)$, which is the probability that learner's belief predicts the label of x to be y. We define the payoff of a learner's policy according to the probability by which each example is selected in the policy. Let $\pi_t^{\mathcal{L}}$ be the policy of learner in interaction t and $u_a(\theta_t^{\mathcal{L}}, \pi_t^{\mathcal{L}})$ denote such payoff. The payoff $u_a(\theta_t^{\mathcal{L}}, \pi_t^{\mathcal{L}})$ is $\sum_x \theta_t^{\mathcal{L}}(y \mid x)$ $\pi_t^{\mathcal{L}}(x)$ where y is the label of x and x goes over all presented examples in interaction t. It is the expected value of the payoff of every labeled example in interaction t computed over the probability distribution of $\pi_t^{\mathcal{L}}$.

The payoff function $u_a(.)$ encourages the learner to build a belief over all the hypotheses that is consistent with and accurately generalizes to the (future) trainer's labeling. Nevertheless, it may lead the learner to build a model that is biased toward a subset of the data and present examples only from that subset. For instance, assume a learner has a belief that accurately predicts labels for a small subset of examples from interaction t onward. It can use a policy that puts 0 probability on the examples outside of this subset and receive maximum payoff for each interaction after t. Such a learner does not have any incentive to learn a model that accurately predicts the labels of examples outside the subset. Thus, all other conditions being the same, policies that provide more varieties of examples may get higher payoff. Moreover, as explained before, the more *representative* and *informative* the presented examples are, the more knowledge the trainer may achieve about the data.

There are multiple methods to measure the coverage and representativeness of samples from a dataset. We use *entropy* of the learner's policy in each interaction to quantify such properties for its examples. Let $u_L(\theta_t^{\mathcal{L}}, \pi_t^{\mathcal{L}})$ denote the total payoff of the learner policy $\pi_t^{\mathcal{L}}$ in interaction t. We have $u_L(\theta_t^{\mathcal{L}}, \pi_t^{\mathcal{L}}) = u_a(\theta_t^{\mathcal{L}}, \pi_t^{\mathcal{L}}) - \gamma \sum_x \pi_t^{\mathcal{L}}(x) \ln \pi_t^{\mathcal{L}}(x)$ where x is an example and $\gamma > 0$ is a real number that sets the balance between two aspects of payoff. The lower the value of γ is, the less exploratory the selected set of examples by the learner are. The parameter γ expresses both the lack of knowledge of the trainer about the data and the lack of confidence of the learner about its belief. The trainer may setup the value of γ according to its background information about the data.

Interactive Learning and Adaptation. Agents leverage the information gained in each interaction to gradually update and improve the accuracy of their beliefs according to some learning method. Improved beliefs in turn guide agents to choose policies with higher payoffs and to learn collaboratively an accurate target model. That is, with accurate beliefs, the trainer provides more effective labelings and the learner presents more informative examples whose labeling may lead to learning an accurate model faster. Each learning method consists of two components: a *prediction model*, P, that updates the agents' belief and a *response model*, R, using which the agent selects its policy based on the updated belief Young (2004). Let the *history* of interactions up to time $t \geq 0$, denoted as h_t, be the sequence of labelings $(x, y)_0 \dots (x, y)_{t-1}$ of interactions 0 to $t-1$. The prediction model of agent i, shown as P^i, is a stochastic mapping from h_t and current belief θ^i to the set of agent's updated beliefs. The response model of agent i, denoted as R^i, is a stochastic mapping from the set of its beliefs to the set of its policies.

Trainer's Prediction & Response Models. Suppose the trainer's belief is $\theta_{t-1}^{\mathcal{T}}$ at interaction $t-1$. This belief is a result of (i) the trainer's prior belief before observing any data and (ii) the samples of data that were presented to the trainer till time $t-1$. Let X_t denote the set of examples presented to the trainer at interaction t. After observing X_t, the trainer updates its belief based on how much X_t accords or contrasts with $\theta_{t-1}^{\mathcal{T}}$. This updated belief of the trainer is a function of $\theta_{t-1}^{\mathcal{T}}$ and previous observed examples $X_1, X_2, \dots, X_t$ as

$\theta_t^{\mathcal{T}} = P^{\mathcal{T}}(\theta_{t-1}^{\mathcal{T}}, \mathbf{X^1}, \mathbf{X^2}, ..., \mathbf{X^t})$. In interaction t, the trainer labels every example $x_t \in X_t$ with labels y_t based on policy $\pi_t^{\mathcal{T}} = R^{\mathcal{T}}(\theta_t^{\mathcal{T}})$.

Learner Prediction & Response Models. The learner uses its predictive model to predict the trainer's current belief. Let the trainer provide a set of labeled examples Y_t in interaction t. Using its prediction model, the learner uses Y_t and updates its belief to match the trainer's belief such that the learner can predict future labeling as accurately as possible. We have $\theta_t^{\mathcal{L}} = P^{\mathcal{L}}(\theta_{t-1}^{\mathcal{L}}, \mathbf{X^t}, \mathbf{Y^t})$. The learner selects and shows examples from the underlying dataset in interaction t using policy $\pi_t^{\mathcal{T}}$. Before showing examples, the learner updates it policy using its response model as $\pi_t^{\mathcal{T}} = R^{\mathcal{L}}(\theta_t^{\mathcal{L}})$.

3. How Do Humans Learn To Train?

Researchers have generally categorized human learning in interactive games as *model-based* and *model-free* learning Young (2004). We have explained model-based learning methods in Section 2. In model-free methods, the agent updates its response strategy without constructing any belief about the state of the game and directly reinforces policies based on the observed payoffs. We believe that model-free methods do not accurately model the learning of human trainers in our setting as there is not any direct observed payoff in interactions, therefore, trainers have to rely on their beliefs to estimate payoff of policies. Thus, we focus on prominent model-based methods.

Fictitious Play. Fictitious play (FP for short) is a fundamental model-based learning method in interactive games. In FP, the agent assumes the other agent's strategy does not change in the course of interaction, i.e., it is stationary. Its belief about the other player is a probability distribution on the other player's actions. The agent updates the probability of each action in its belief using the observed empirical frequencies of that action so far. In each interaction, it picks a strategy that delivers the highest payoff assuming the other players performs the action with the highest probability according to the belief. The agent may have a prior belief before starting the interaction, which represents its domain knowledge. In the absence of domain knowledge, the agent may use an uninformative uniform prior. As FP is simple and requires relatively less degree of rationality, information, and computational resources, it is a popular method to model human learning.

Bayesian Learning. In Bayesian learning, the agent maintains a prior distribution over the likelihood of performing actions by the other player. After each interaction, it observes the other agent's actions and modifies its belief using Bayesian updating. As opposed to FP where a belief is a probability distribution over actions of the other player, a belief in Bayesian learning is a distribution over the probabilities (parameters of probability distributions) of actions of the other player. Similar to FP, in each interaction, the agents picks the action or strategy that delivers the highest payoff given its belief. Given some mild assumptions on the prior belief, Bayesian learning is equivalent to FP Fudenberg and Levine (1998). Thus, in this paper, we use FP and Bayesian in this paper interchangeably.

Hypothesis Testing. In hypothesis testing, the agent starts with an initial hypothesis and belief on how the other agent will act in the interaction. The agent frequently evaluates the performance of its belief and rejects the current one if it does *not* explain sufficient amount of recent data with some tolerance. If the current hypothesis is rejected, the agent picks another hypothesis based on its relative performance on the recent data. This is a popular model in both economics and cognitive psychology Young (2004).

User Study. The results of our user study (Section A) indicate that humans use FP (Bayesian) significantly more often than hypothesis testing to train a model.

4. Learning Methods for Learner

Stochastic Best Response. Assume that the learner follows FP as its prediction model. Consider the response strategy that selects a policy in interaction t that picks example x according to the probability $\frac{e^{u_a(\theta_t^{\mathcal{L}},x)/\gamma}}{\sum_{x' \in D} e^{u_a(\theta_t^{\mathcal{L}},x')/\gamma}}$ where x and x' are examples, D is the dataset, and γ is the hyper-parameter defined in Section 2. This policy maximizes the payoff function of the learner in interaction t given its belief Fudenberg and Levine (1998). It establishes a balance between selecting examples with higher payoff of $u_a(.)$ and showing an informative and diverse set of examples via stochastic choosing. Thus, it may present a more representative set of examples than the ones current deterministic policies show to the trainer, ultimately addressing their shortcoming. We call this response strategy *stochastic best response* (*stochastic best*). We denote the learning method that uses FP for prediction and stochastic best for response strategy as (FP, Stochastic Best).

Proposition 1 *If the trainer and learner use learning methods (FP, Best) and (FP, Stochastic Best), respectively, the empirical behaviour of the game converges to an equilibrium.*

Stochastic Uncertainty Sampling. Stochastic best strategy is rather short-sighted as it aims at collecting immediate reward by presenting examples that it is confident about their labels. This may significantly prolong the number of interactions for learning the accurate belief. To address this issue, the learner may use active learning heuristics, such as uncertainty sampling, to pick examples while preserving the stochastic nature of these policies. That is, it may replace the term $u_a(\theta_t^{\mathcal{L}}, x)$ in the payoff function by the value of uncertainty of x given its belief. We call this response strategy *stochastic uncertainty sampling* (*stochastic uncertainty*). It approximates uncertainty sampling if γ is almost zero.
Convergence to Equilibria. Given a pair of learning methods for the trainer and learner, we would like to know whether the interaction converges to some equilibrium. Let Φ_t^i denote the empirical frequencies, i.e., empirical distribution, of actions performed by agent i up to and including interaction t. Each member of Φ_t^i, denoted as $\Phi_t^i(x)$, is the observed number of occurrences of action x in interactions up to and including time t normalized by t. Let π^i is a policy of agent i.

Definition 2 *Empirical behaviour of agent i converges to π^i as $t \to \infty$ if Φ_t^i converges to π^i almost surely ($P(\lim_{t\to\infty} \Phi_t^i = \pi^i) = 1$) Young (2004).*

Intuitively speaking, if we compute empirical distribution of the realized actions of an agent in all interactions, we find that it converges to a fixed policy in the long run. The *empirical behaviour of a game* converges to an equilibrium if the empirical behaviours of both agents converge to policies $\pi^{\mathcal{T}}$ and $\pi^{\mathcal{L}}$ such that ($\pi^{\mathcal{T}}$, $\pi^{\mathcal{L}}$) is an equilibrium of the game. Let $\mathbf{L}^i$ denote the learning method of agent i. The *empirical behaviour of a learning scheme* $(\mathbf{L}^{\mathcal{T}}, \mathbf{L}^{\mathcal{L}})$ converges to an equilibrium if the empirical behaviour of every game in which the trainer and learner follow $\mathbf{L}^{\mathcal{T}}$ and $\mathbf{L}^{\mathcal{L}}$, respectively, converge to an equilibrium.
Empirical Evaluation. We evaluate the proposed methods through extensive empirical studies over real-world data in Section C. Our results show that the proposed algorithms significantly outperform current active learning algorithms.

References

Ziawasch Abedjan, Cuneyt Gurcan Akcora, Mourad Ouzzani, Paolo Papotti, and Michael Stonebraker. Temporal rules discovery for web data cleaning. *Proc. VLDB Endow.*, 9 (4):336–347, 2015. doi: 10.14778/2856318.2856328. URL `http://www.vldb.org/pvldb/vol9/p336-abedjan.pdf`.

Ziawasch Abedjan, Xu Chu, Dong Deng, Raul Castro Fernandez, Ihab F. Ilyas, Mourad Ouzzani, Paolo Papotti, Michael Stonebraker, and Nan Tang. Detecting data errors: Where are we and what needs to be done? *Proc. VLDB Endow.*, 9(12):993–1004, 2016.

Charu C. Aggarwal, Xiangnan Kong, Quanquan Gu, Jiawei Han, and Philip S. Yu. Active learning: A survey. In Charu C. Aggarwal, editor, *Data Classification: Algorithms and Applications*, pages 571–606. CRC Press, 2014. URL `http://www.crcnetbase.com/doi/abs/10.1201/b17320-23`.

Patricia C. Arocena, Boris Glavic, Giansalvatore Mecca, Renée J. Miller, Paolo Papotti, and Donatello Santoro. Messing up with bart: Error generation for evaluating data-cleaning algorithms. *Proc. VLDB Endow.*, 9(2):36–47, oct 2015. ISSN 2150-8097. doi: 10.14778/2850578.2850579. URL `https://doi.org/10.14778/2850578.2850579`.

Jürgen Bernard, Matthias Zeppelzauer, Michael Sedlmair, and Wolfgang Aigner. Vial: A unified process for visual interactive labeling. *Vis. Comput.*, 34(9):1189–1207, sep 2018a. ISSN 0178-2789. doi: 10.1007/s00371-018-1500-3. URL `https://doi.org/10.1007/s00371-018-1500-3`.

Jürgen Bernard, Marco Hutter, Matthias Zeppelzauer, Dieter Fellner, and Michael Sedlmair. Comparing visual-interactive labeling with active learning: An experimental study. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):298–308, 2018b. doi: 10.1109/TVCG.2017.2744818.

Jürgen Bernard, Matthias Zeppelzauer, Markus Lehmann, Martin Müller, and Michael Sedlmair. Towards user-centered active learning algorithms. *Computer Graphics Forum*, 37(3): 121–132, 2018c. doi: https://doi.org/10.1111/cgf.13406. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13406`.

Xu Chu, Ihab F. Ilyas, and Paolo Papotti. Holistic data cleaning: Putting violations into context. In *ICDE*, pages 458–469. IEEE Computer Society, 2013.

Kyriaki Dimitriadou, Olga Papaemmanouil, and Yanlei Diao. Explore-by-example: an automatic query steering framework for interactive data exploration. In Curtis E. Dyreson, Feifei Li, and M. Tamer Özsu, editors, *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 517–528. ACM, 2014. doi: 10.1145/2588555.2610523. URL `https://doi.org/10.1145/2588555.2610523`.

Wenfei Fan. Dependencies revisited for improving data quality. In Maurizio Lenzerini and Domenico Lembo, editors, *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2008, June 9-11, 2008,* *Vancouver, BC, Canada*, pages 159–170. ACM, 2008. doi: 10.1145/1376916.1376940. URL `https://doi.org/10.1145/1376916.1376940`.

Wenfei Fan and Floris Geerts. *Foundations of Data Quality Management.* Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2012. doi: 10.2200/S00439ED1V01Y201207DTM030. URL `https://doi.org/10.2200/S00439ED1V01Y201207DTM030`.

Drew Fudenberg and David Levine. *The Theory of Learning in Games.* MIT Press, 1998.

Daniel Golovin, Andreas Krause, and Debajyoti Ray. Near-optimal bayesian active learning with noisy observations. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1*, NIPS'10, page 766–774, Red Hook, NY, USA, 2010. Curran Associates Inc.

Jian He, Enzo Veltri, Donatello Santoro, Guoliang Li, Giansalvatore Mecca, Paolo Papotti, and Nan Tang. Interactive and deterministic data cleaning. In *SIGMOD*, pages 893–907. ACM, 2016.

Alireza Heidari, Joshua McGrath, Ihab F. Ilyas, and Theodoros Rekatsinas. Holodetect: Few-shot learning for error detection. In *SIGMOD*, pages 829–846. ACM, 2019.

Josef Hofbauer and William H. Sandholm. On the global convergence of stochastic fictitious play. *Econometrica*, 70(6):2265–2294, 2002. ISSN 00129682, 14680262. URL `http://www.jstor.org/stable/3081987`.

Benjamin Höferlin, Rudolf Netzel, Markus Höferlin, Daniel Weiskopf, and Gunther Heidemann. Inter-active learning of ad-hoc classifiers for video visual analytics. *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 23–32, 2012.

Ykä Huhtala, Juha Kärkkäinen, Pasi Porkka, and Hannu Toivonen. TANE: an efficient algorithm for discovering functional and approximate dependencies. *Comput. J.*, 42(2): 100–111, 1999.

Ihab F. Ilyas, Volker Markl, Peter Haas, Paul Brown, and Ashraf Aboulnaga. Cords: Automatic discovery of correlations and soft functional dependencies. In *SIGMOD*, 2004.

Jyrki Kivinen and Heikki Mannila. Approximate dependency inference from relations. In Joachim Biskup and Richard Hull, editors, *ICDT*, volume 646 of *Lecture Notes in Computer Science*, pages 86–98. Springer, 1992.

Rui Li, Rui Guo, Zhenquan Xu, and Wei Feng. A prefetching model based on access popularity for geospatial data in a cluster-based caching system. *International Journal of Geographical Information Science*, 26(10):1831–1844, 2012.

Ester Livshits, Benny Kimelfeld, and Sudeepa Roy. Computing optimal repairs for functional dependencies. *ACM Trans. Database Syst.*, 45(1), 2020. ISSN 0362-5915. doi: 10.1145/3360904. URL `https://doi.org/10.1145/3360904`.

Mohammad Mahdavi, Ziawasch Abedjan, Raul Castro Fernandez, Samuel Madden, Mourad Ouzzani, Michael Stonebraker, and Nan Tang. Raha: A configuration-free error detection system. In *SIGMOD*, pages 865–882. ACM, 2019.

Y Niv. Reinforcement learning in the brain. *The Journal of Mathematical Psychology*, 53 (3):139–154, 2009a.

Yael Niv. The neuroscience of reinforcement learning. In *ICML*, 2009b.

Theodoros Rekatsinas, Xu Chu, Ihab F. Ilyas, and Christopher Ré. Holoclean: Holistic data repairs with probabilistic inference. *Proc. VLDB Endow.*, 10(11):1190–1201, 2017.

Esther Rolf, Nikolay Malkin, Alexandros Graikos, Ana Jojic, Caleb Robinson, and Nebojsa Jojic. Resolving label uncertainty with implicit posterior models. In James Cussens and Kun Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 1707–1717. PMLR, 01–05 Aug 2022. URL `https://proceedings.mlr.press/v180/rolf22a.html`.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009. URL `http://axon.cs.byu.edu/~martinez/classes/778/Papers/settles.activelearning.pdf`.

Burr Settles. *Active Learning*. Morgan & Claypool Publishers, 2012. ISBN 1608457257.

Pannaga Shivaswamy and Thorsten Joachims. Coactive learning. *J. Artif. Int. Res.*, 53(1): 1–40, may 2015. ISSN 1076-9757.

Saravanan Thirumuruganathan, Laure Berti-Équille, Mourad Ouzzani, Jorge-Arnulfo Quiané-Ruiz, and Nan Tang. Uguide: User-guided discovery of fd-detectable errors. In *, SIGMOD*, pages 1385–1397. ACM, 2017. doi: 10.1145/3035918.3064024. URL `https://doi.org/10.1145/3035918.3064024`.

Paroma Varma and Christopher Ré. Snuba: Automating weak supervision to label training data. *Proc. VLDB Endow.*, 12(3):223–236, nov 2018. ISSN 2150-8097. doi: 10.14778/3291264.3291268. URL `https://doi.org/10.14778/3291264.3291268`.

Mohamed Yakout, Ahmed K. Elmagarmid, Jennifer Neville, Mourad Ouzzani, and Ihab F. Ilyas. Guided data repair. *Proc. VLDB Endow.*, 4(5):279?289, February 2011. doi: 10.14778/1952376.1952378.

Songbai Yan, Kamalika Chaudhuri, and Tara Javidi. Active learning from imperfect labelers. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *NIPS*, pages 2128–2136, 2016.

H Peyton Young. *Strategic learning and its limits*. OUP Oxford, 2004.

Chicheng Zhang and Kamalika Chaudhuri. Active learning from weak and strong labelers. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *NIPS*, pages 703–711, 2015.

Ugur Çetintemel, Mitch Cherniack, Justin DeBrabant, Yanlei Diao, Kyriaki Dimitriadou, Alexander Kalinin, Olga Papaemmanouil, and Stanley B. Zdonik. Query steering for interactive data exploration. In *CIDR*, 2013.

Appendix A. User Study

A.1 Use Case: Approximate FD Discovery

Our approach is independent from the target application and from the internals of the learning system as long as such system, i.e., learner, exposes examples and consumes labels. For our user study and empirical evaluation, we illustrate our approach with a system that aims at learning approximate functional dependencies (FDs) Kivinen and Mannila (1992).

Approximate FD Discovery over Real-world Data. We focus on the challenging case of a learner finding approximate FDs over *a dataset that may contain dirty tuples*. If the dataset is completely clean and does not contain any dirty/erroneous tuple, its set of approximate FDs can be learned with an unsupervised method Huhtala et al. (1999); Ilyas et al. (2004). Nonetheless, real-world datasets often contain dirty tuples. In this case, the learner requires some supervision to accurately learn approximate FDs Abedjan et al. (2015). This requires user input to find out whether a violation of an FD in the data is due to some erroneous tuple. In other words, users should provide the learner with sufficiently many labeled tuples, i.e., annotated as clean or dirty.

Users' Input. In this kind of interaction, the user only annotates examples, without providing (approximate) FDs explicitly. Users provide labels according to their beliefs on approximate FDs, which is not observed by the learner directly. We conduct a user study (Section 3) to analyze how users, i.e., trainers, learn about FDs while providing annotated examples interactively. To evaluate different models of users' learning and how their beliefs improve over time, we gather the ground truth on users' beliefs. Thus, in addition to annotated examples, we ask participants to provide the set of FDs they believe to be the most accurate in each interaction. This is limited to our user study with the aforementioned goal. In our framework and algorithms, we assume that trainers provide the learner only with annotated examples.

Applications of Approximate FDs. This learned approximate FDs can be used for detecting errors in unlabeled or future tuples Fan (2008); Fan and Geerts (2012); Yakout et al. (2011); Abedjan et al. (2016); Chu et al. (2013); Rekatsinas et al. (2017); Livshits et al. (2020); Abedjan et al. (2015). In this paper, we focus on learning and discovering approximate FDs. The study of new algorithms to make the best use of these learned models is out of the scope of this paper.

Why Approximate FDs. For our learner module, we focus on declarative rules, like FDs, rather than a "black-box" ML method for two main reasons. First, many popular models in data management, e.g., data cleaning systems, use declarative rules. Second, as explained above, to conduct a user study and investigate the connection between users beliefs and the labels they provide, one may have to solicit the participants to provide both labeled examples and their belief about the accurate model in each interaction. It is more convenient for humans to describe their beliefs in form of (simple) declarative rules, such as FDs, rather than other models.

	Player	Team	City	Role	Apps
t_1	Carter	Lakers	L.A.	C	4
t_2	Jordan	Lakers	Chicago	PF	4
t_3	Smith	Bulls	Chicago	PF	4
t_4	Black	Bulls	Chicago	C	3
t_5	Miller	Clippers	L.A.	PG	3

Table 1: Sample instance of a dataset D.

Approximate Functional Dependencies Given a relation r over a schema R with attribute sets $X, Y \subseteq R$. We represent the projection of a tuple t to a set of attributes X as $t[X]$. We denote with $X \to Y$ an FD with left-hand side (LHS) X and right-hand side (RHS) Y if we have $t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$ for all pairs of distinct tuples $t_1, t_2 \in r$. An $X \to A$ is minimal if no subset of X determines attribute A. We focus our study to FDs that are minimal, non trivial, i.e., $X \cap Y = \varnothing$, and normalized, i.e., the RHS is a single attribute.

Due to the presence of errors, in most real datasets FDs are rarely completely satisfied by the data. *Exact* FDs can therefore be relaxed to allow a certain degree of violation. We refer to such relaxed FDs as *approximate* FDs. To quantify the degree of approximation, we use the g_1 measure in its scaled version, which represents the percentage of violating pairs of tuples.

$$g_1(X \to A, r) = \frac{|\{(t_1, t_2)|t_1, t_2 \in r, t_1[X] = t_2[X], t_1[A] \neq t_2[A]\}|}{|r^2|}$$

Example 1 *Consider dataset D in Table 1. Let f_1 be the approximate FD Team $\to$ City. The pair of tuples t_3 and t_4 satisfy the FD, but tuples t_1, t_2 do not. The g_1 for f_1 over D is therefore $\frac{1}{25} = 0.04$.*

FD discovery is the problem of finding all minimal, nontrivial functional dependencies that hold in a given relation.

Detecting Errors Every pair of tuples involved in G_1 represents a *violation* of the given FD. In the data cleaning literature Arocena et al. (2015), violations are also identified at a finer granularity, i.e., at the cell value level. A violation is defined by the set of attribute cells for X and Y over the two violating tuples. Given a relation r, the set of all violating cells that can be determined through FDs over T is denoted by C_v.

Given one or more violations, there is plethora of algorithms that, given a set of FDs, try to identify the tuples (or cells) that contain erroneous values Chu et al. (2013); Rekatsinas et al. (2017); Abedjan et al. (2016); Livshits et al. (2020). As we focus on approximate FDs, any algorithm for error detection in the presence of approximate FDs can be used in our framework. Intuitively, for an FD f over r and a pair tuples in r that satisfy f, we convert the scaled g_1 measure m of f into the probability of being dirty for the two tuples. For a violating pair of tuples, we consider $1 - m$ as the probability of being dirty for the two tuples.

Example 2 *Consider again dataset D in Table 1. Given approximate FD f_1 over Team and City, t_1 and t_2 are in violation as they satisfy f_1 LHS and do not satisfy its RHS. They*

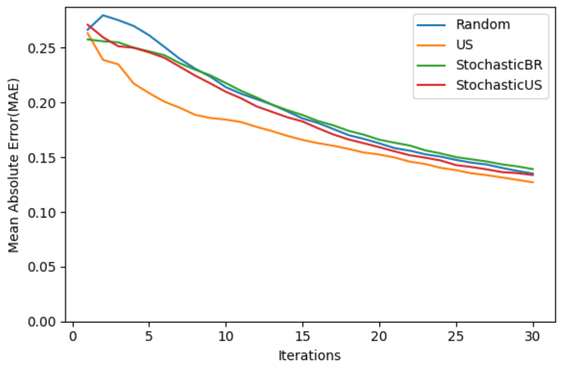


Figure 1: Mean Absolute Error between Trainer and Learner models for *OMDB* dataset with $\approx 10\%$ violations, trainer's prior model=Random, learner's prior model=Data-estimate

are assigned a 0.96 probability of being dirty. Indeed, at least one of the tuples contains an erroneous cell value, but this is not identified by the FD itself.

A.2 User Study Setup

Users & Interface Our user study involves a group of 20 undergraduate and graduate students with different background in Computer Science. We have implemented a user interface that informs users about FDs, FD violations, and tests their knowledge. Then, the user is presented with the dataset schema and asked what they believe is the FD(s) that holds over the dataset with the fewest exceptions. We use this model as the prior belief of the users about the dataset in the methods described in Section **??**. The user also has the option of indicating that they are not sure what the most accurate initial model is, in which case we use a uniform prior. Afterwards, the interface iteratively presents examples to the users for marking FD violations.

In each iteration, our system shows a random sample of ten (10) tuples from the dataset to the user. The user can mark violations of their hypothesized FD(s) in the presented examples according to their belief in that interaction. We instructed the users to label errors detectable by FDs. After marking violations in an interaction, the users specify their current hypothesized FD(s). Hence, we have direct access to the user belief in each iteration during user study, which enables us to measure the accuracy of human learning schemes effectively - such FDs are not fed to the learner. Each user must interact for at least 9 iterations and can continue up to 15 iterations for each scenario.

Our goal is to model how user interactively learn the FDs that hold over the data. It is time-consuming to ask users to specify models with multiple FDs in every interaction, this could discourage users from participating in the study. Hence, we ask users to specify explicitly the FD that they deem is hold over the observed data so far most accurately and label the presented samples accordingly.

Datasets & Scenarios We use two real-world datasets: *AIRPORT* (Alaska Airport Data from *www.kaggle.com/datasets/, jamestollefson/alaskaairfields*) describes airports, heliports, and seaplane bases throughout the U.S. state of Alaska, while *OMDB* (Open Movie Database from *www.omdbapi.com*) contains information about English-language movies and TV shows.

We use these datasets to cover both relatively familiar, i.e., movies, and unfamiliar domains to users in the study. We design five scenarios based on these two datasets whose information is shown in Table 2. In each scenario, we ask the participants to identify the FD(s) that hold over the underlying data with the fewest violations (exceptions), i.e., *target FD(s)*. Table 2 illustrates examples of other FDs in each scenario, i.e., *alternative FD(s)*. The participants may believe some of these FDs may hold over the dataset with fewest exceptions, e.g., given the schema and an initial looks at the data. The exact attributes present in each dataset vary slightly by scenario.

#	**Domain**	**Attributes**	**FDs**
1	Airport	facilityname, type, manager	**Target**: *(facilityname, type)* $\Rightarrow$ *manager* **Alternative**: *facilityname* $\Rightarrow$ *(type, manager)*
2	Airport	sitenumber, facilityname, owner, manager,	**Target**: *sitenumber* $\Rightarrow$ *(facilityname, owner, manager)* **Alternative**: *facilityname* $\Rightarrow$ *(sitenumber, owner, manager)*
3	Airport	facilityname, owner, manager,	**Target**: *manager* $\Rightarrow$ *owner* **Alternative**: *facilityname* $\Rightarrow$ *(owner, manager)*
4	OMDB	title, year genre, type	**Target**: *(title, year)* $\Rightarrow$ *(type, genre)* **Alternative**: *title* $\Rightarrow$ *(year, type, genre)*
5	OMDB	title, rating, type	**Target**: *rating* $\Rightarrow$ *type* **Alternative**: *title* $\Rightarrow$ *(rating, type)*

Table 2: Scenarios used in the users study

We introduce violations to each dataset in every scenario with an error generation tool that scrambles values w.r.t. the target FD Arocena et al. (2015). The *violation ratio* of $\frac{m}{n}$ is to introduce n violations in every alternative FD per each m violations in the target ones. We use ratios of $\frac{1}{3}$ and $\frac{2}{3}$ for the first three and last two scenarios, respectively. We use different ratios to investigate the impact of the relative quantity of violations on human learning. The smaller the violation ratio is, the easier it may be for the participant to pinpoint the target FD(s). We use the smaller ratios for the scenarios from the *AIRPORT* domain as it is relatively less familiar for the participants.

We randomize both the order that the scenarios are presented to participants and the arrangement of the attributes in the samples presented to participants for each scenario. This ensures that our findings are agnostic to order of presentation.

Configuration of Learning Methods We use 20% of our collected information to tune the hyper-parameters of different methods as follows. Given the prior belief of the user about the most accurate FD, we build a prior beta distribution for that FD to be used as the prior in the FP and Bayesian method. We set its mean to $\epsilon = 0.85$. We call FD $X \rightarrow Z$ a *superset* of $XY \rightarrow Z$ where X, Y, Z are non-empty sets of attributes. FD $XY \rightarrow Z$ is a *sub-set* of $X \rightarrow Z$. If FD f_1 is a super-set of FD f_2, f_2 is implied by f_1. If f_1 is a sub-set of f_2, they are semantically close. Thus, we use two types of prior configurations for evaluating the learning methods. In the first one, we set the mean value of the prior distributions for all FDs other than the one specified by the user to the same value, which is set to 0.15. In the second one, we treat the prior distributions of FDs that are superset or subset of the one specified by the user differently than other FDs. We set their mean values to 0.8. We use the same prior for non-minimal FDs entered by the users, e.g., $XY \rightarrow YZ$. The standard deviation of all prior distributions are set to the same value of 0.05.

We set the initial values of α and β using the mean and variance equations of beta distributions to $\mu = \frac{\alpha}{\alpha+\beta}$ and $\sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$. For our hypothesis testing learning method, we use the same model space and initial model setting as the one used in FP and Bayesian method. We set the testing frequency to every interaction. In our experiments, hypothesis testing have performed best over the majority of our training data when the users test their hypothesis using the samples presented in the preceding interaction. If the user rejects the current model given the aforementioned subset of tuples, they pick the model that performs the best over this subset.

Evaluation Metric The aim of all learning methods is to predict the user specified FD accurately based on the labeling provided by the user. To measure the accuracy of prediction for each method, we first sort the top-k predicted FDs by the method in each interaction based on their confidence. Let p be the position of the user specified FD, i.e., ground truth, in the list. The **Reciprocal Rank** (RR) of the method is $\frac{1}{p}$. We use the mean of RR values, i.e., **Mean Reciprocal Rank** (MRR), across all interactions to measure their overall accuracy in each scenario. WE set k to 5 in our experiments.

We also report and measure the accuracy of each method by also considering the predicted FDs that are subset/ supersets of the ground truth ones. We penalize matches of subset/ superset FDs. Let $c(f)$ and c_g denote the set of compliant tuples with FD f and the set of clean tuples in the ground truth labeled data, respectively. The *precision* and *recall* of an FD f are $\frac{|c(f) \cap c_g|}{|c(f)|}$ and $\frac{|c(f)|}{|c_g|}$, respectively. *F1 score* of FD f is the harmonic mean of its precision and recall. We discount matches of subset/ superset FDs using their *F1 score* differences with the user supplied one. In our results, we distinguish the cases where we consider subset/superset matches as well as exact matches by including a "+" in their names.

A.3 Results

Users' Learning Activity & Labeling Errors To quantify the extent of learning activities and labeling errors, we report the degree by which the f1-score of the user's hypotheses changes between iterations to determine whether these changes are due to noise or significant changes to users' beliefs. Indeed, as users label tuples in each interaction according to their current belief, when users hold an inaccurate belief about the data, e.g., wrong FDs, they provide erroneous labels. We can see in Table 3 that in all but one scenario, the average change in f1-score is relatively large, especially when considering that a change of 0.33 is the difference between an FD that explains only $\frac{2}{3}$ of the violations in the dataset and an FD that explains almost all of the violations. This suggests that changes in the user's hypotheses from iteration to iteration are likely not due to simple noise but is rather due to the user updating their hypothesis of which FD best explains the violations in the data.

Scenario #	Average change in $\mathbf{f_1}$-score
1	0.1144
2	0.3280
3	0.2301
4	0.2843
5	0.1767

Table 3: Average $\mathbf{f_1}$-score change between any two rounds of user labeling, grouped by scenario

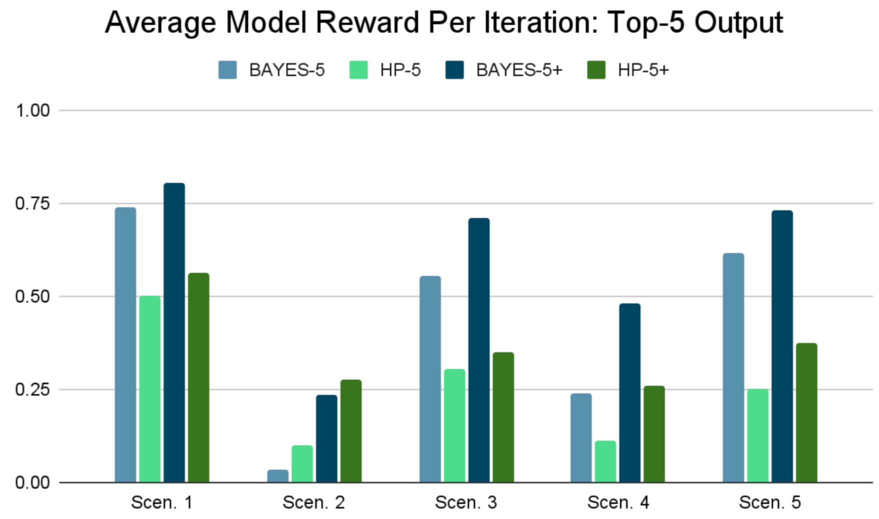


Figure 2: MRR computed over all interactions for each learning model with $k = 5$

Learning Models

Figure 2 shows the accuracy of different methods in predicting changes in the users' hypotheses over all interactions. Results show that in all but one scenario the Bayesian (FP) model significantly outperforms hypothesis testing in modeling participants' behavior. This trend holds true as the metric of success is loosened to accept subsets or supersets of the user's hypothesis. In most scenarios and interactions, the user's hypothesis is among the top-1 or 2 returned FDs by the Bayesian model. Given the size of the set of possible FDs, this shows that Bayesian (FP) model accurately models users' learning in most cases. This confirms the popularity of FP in empirical game theory and the general belief that as it is simple and does not require much (computational) resources, it accurately models learning behaviour in many settings. We observe a similar trend when grouping predictions based on participants: Bayesian (FP) model significantly outperform hypothesis testing for all our participants except for two.

One important exception to the general trend is scenario 2 in which none of the available learning models are able to accurately predict participant's belief. While we have observed some degree of non-monotone learning behaviour in other scenarios, participants show a significantly less monotone learning in scenario 2 as they often moved from more accurate beliefs to less accurate ones. This may be due to the fact that this scenario is rather more difficult than others. Since current models for human learning expect some level of monotonicity in learning behaviour, they may not be able to predict human's learning in such settings. One may further improve these models by considering the probability of noise in decision making Young (2004). As Bayesian (FP) model accurately models human learning in our study in most cases, we will use it as the model for trainer's learning in our framework.

Appendix B. Proofs

Proof for Proposition 1.

Since the training game is a game of identical interest, it is also a potential game Young (2004). As best response is a special case of stochastic best response, it follows from Theorem 6.1 in Hofbauer and Sandholm (2002).

Appendix C. Empirical Study

We evaluate and compare the number of interactions and samples each method in Section 4 required to learn a common belief with a learning trainer in the average-case.

C.1 Experimental Setting

Datasets & Models. We use four datasets: *OMDB* and *Airport* from our user study in Section 3, *Tax* and *Hospital* from the error detection literature Mahdavi et al. (2019); Chu et al. (2013); Arocena et al. (2015). *Hospital* is a real-world dataset with 19 attributes and six exact FDs, while *Tax* is a synthetic dataset with 15 attributes and four exact FDs. As opposed to learning one target model in Section 3, in this experiment the learner learns a model for 38 approximate FDs for each dataset, i.e., a distribution over the confidence of each FD. Each FD has at most four attributes. The goal for the learner is to reach an agreement with the model of the learning trainer as fast as possible, as shown in Figure **??**.
State-of-the-art & Proposed Methods. We compare our proposed methods with the state-of-the-art active learning technique of *Uncertainty Sampling (US).* As explained before, US is widely used in current active learning settings and assume the trainers have a fixed belief and do *not* learn during labeling. We also implement the baseline method of *Fixed Random Sampling (Random)* method that picks examples uniformly at random and present them to the trainer for labeling. We implement our proposed methods *Stochastic Best Response (StochasticBR)* (Section **??**) and *Stochastic Uncertainty Sampling (StochasticUS)* (Section **??**). For measure of uncertainty in both US and StochasticUS, we use entropy as $entropy(x, \theta_t) = -p_{\theta_t}(x)\log(p_{\theta_t}(x)) - (1-x)\log(1-p_{\theta_t}(x))$, where $p_{\theta_t}(x)$ is probability of x being a clean tuple based on belief θ_t. The value of γ for the *StochasticBR* and *StochasticUS* is set to be 0.5 in all experiments. This value reduces the greediness in sampling compared to the softmax function where $\gamma = 1$. As explained in Section A.1, FD violations are defined and validated over pairs of tuples. Hence, we have modified all response and sampling methods to select a pair of tuples instead of a single one.
Trainer's Method. Based on our findings in Section 3, we simulate the trainer's learning using FP (Bayesian).
Learner's & Trainer's Priors. We test a set of prior beliefs (*Uniform-d, Random, and Data-estimate*) for the trainer to measure the sampling methods' sensitivity of convergence to the different types of prior beliefs. In *Uniform-d*, the confidence of all the FDs in the

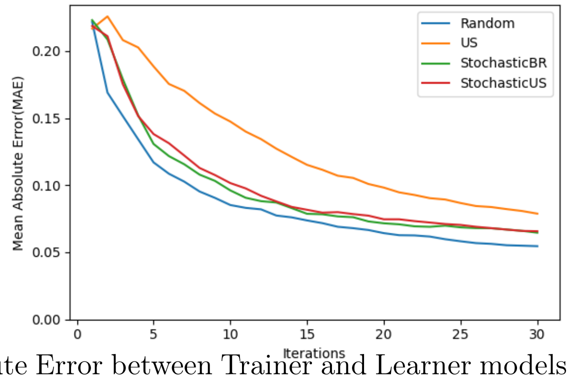


Figure 3: Mean Absolute Error between Trainer and Learner models for *OMDB* with $\approx 10\%$ violation, trainer's prior model=Random and learner's prior model=Uniform-0.9

hypothesis space is initialized to the same value $d \in [0, 1]$. In *Random*, the confidence of each FD is sampled randomly from $[0, 1]$. For *Data-estimate*, the prior is set to the average confidence computed based on the initial unlabeled dataset. This prior represents the case where the learner computes its prior by treating the unlabeled dataset to be completely clean, which is often used in practice.

Degrees of FD Violation. We evaluate the aforementioned methods by injecting different degrees of violations ($< 35\%$) for FDs in the original data. For every dataset, we identify a subset of the tuples so that the fraction of tuple pairs that are violations of the FDs in this sampled dataset is equal to the desired degrees of violations. The aim is to study the influence of the amount of FD violations on the performance of different response and sampling methods.

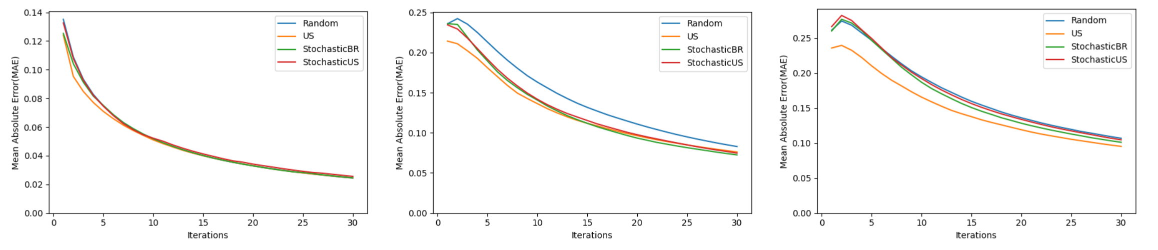
Interactions. At each iteration, Bayesian models corresponding to the learner and trainer are initialized from the set of prior beliefs. Based on the sampling method, the trainer serves a sample data of size $k = 10$. Upon receiving a sample, the trainer updates its model and labels the data based on its updated model. Using the violation labels from the trainer, the learner updates its model accordingly. This interaction continues for $N = 30$ iterations.

Evaluation Metrics: We evaluate the aforementioned methods in terms of **their speed of convergence** and **effectiveness**. We measure the speed of convergence by computing how close the model of the learner is to the trainer's model at each iteration. For this, we use the *mean absolute error* (MAE) of confidence of the models over the hypothesis space. MAE measures the average difference between the confidence of the trainer model and the learner model over the FDs in the hypothesis space. This value can range from 0 to 1; lower value implying closeness between the models of these two agents. To evaluate the effectiveness of each method in error detection, we report *F1 score* for the learner's model in each interaction. We separate 30% of each dataset as the test set and compute F1 score using these test sets.

C.2 Experimental Results

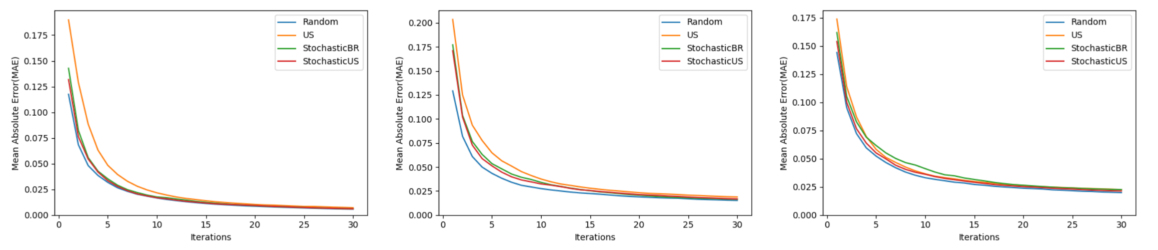
Convergence In this section, we analyze convergence of different methods.

Sampling methods and Prior Models. *Fixed Random Sampling* method is the simplest sampling method as it completely ignores the gained information about the data and does



(a) Airport (b) Hospital (c) Tax

Figure 4: MAE between Trainer and Learner Model, trainer's prior=Random, learner's prior=Data-estimate with ≈ 20% violation



(a) Airport (b) Hospital (c) Tax

Figure 5: MAE between Trainer and Learner Model, trainer's prior=Random, learner's prior=Uniform-0.9 with ≈ 20% violation

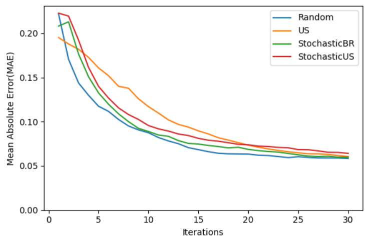
not form a model. At the other extreme, *Uncertainty Sampling* fully relies on its current model and samples tuples greedily based on that.

We consider the case where Trainer's and Learner's prior models are different. Figures 1 and 4 show the results for the case with the trainer starting with a Random prior model and learner's with a prior based on Data-estimate. Results show that for all datasets, while *Random Sampling* reduces the mean error on average, its convergence is slower than with *Uncertainty Sampling*. Results for the stochastic sampling methods lie between these two methods because, while they exploit the model, they use stochastic sampling instead of acting greedily.

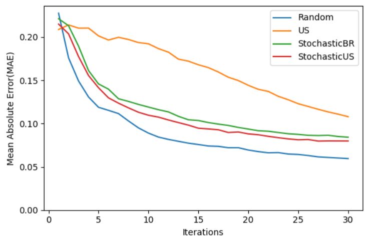
Figures 3 and 5 show that results change significantly when the learner's prior is not informed about the data at hand, i.e., when learner uses a uniform distribution. In this case, instead of helping, the wrong model can hurt the performance w.r.t. Random.

We conclude that, *Uncertainty Sampling* performs best when it is aware of the dataset at hand, which is the standard setting in practice. Otherwise, in the extreme case that the system does not have access to data, *Random Sampling* works best. Considering both scenarios, *Stochastic Best Response* and *Stochastic Uncertainty Sampling* work best on average compared to *Random Sampling* and *Uncertainty Sampling*.

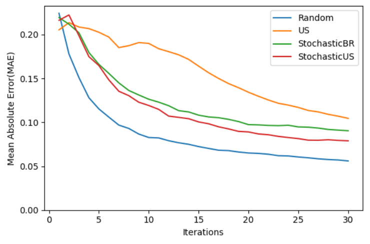
Degree of Violation in Data. We evaluate how the performance of different methods change by increasing the degree of violation in the data. Figure 6 shows that if the model's priors are different, then the increment of violations exacerbates their performance. Our experiments for the cases where the trainer and learner models are in agreement, not shown due to lack of space, indicate that an increment in violations does not impact the results



(a) Degree of violation ≈ 5%

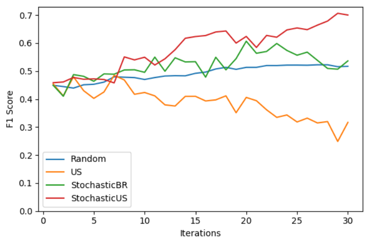


(b) Degree of violation ≈ 15%

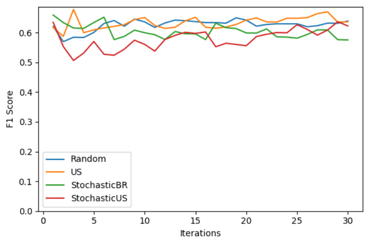


(c) Degree of violation ≈ 25%

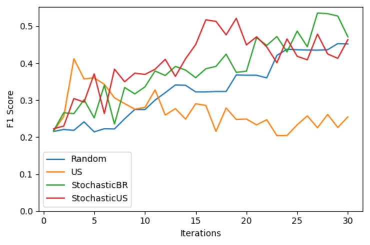
Figure 6: MAE between Trainer and Learner Model for *OMDB*, trainer's prior =Random, learner's prior=Uniform-0.9



(a) OMDB



(b) Hospital



(c) Tax

Figure 7: Average F1 Score of the labeling of Learner model, trainer's prior=Random, learner's prior=Random with ≈ 20% violation

considerably. The results for other datasets demonstrate a similar trend and not shown due to the lack of space.

Accuracy As explained in Section C.1, we compare the error detection accuracy of our proposed methods, *Stochastic Best Response (StochasticBR)* and *Stochastic Uncertainty Sampling (StochasticUS)*, with the popular method of *Uncertainty Sampling (US)* in active learning and the baseline method of *Fixed Random Sampling (Random)*. Figures 7 shows the F1 scores of different methods over *OMDB*, *Hospital*, and *Tax* datasets for random priors of the learner and trainer. Overall, our proposed methods outperform or deliver the same level of F1 score as *Uncertainty Sampling* and *Fixed Random Sampling*. The relatively high values of F1 score for *Fixed Random Sampling* is mainly due to its high recall. Since it selects training examples uniformly at random, it provides the learner with an accurate overall understanding of the dataset. But, it generally delivers lower precision than other methods. This is because it does not provide the learner with sufficiently informative training examples in the reported interactions. *Uncertainty Sampling* delivers a lower recall and precision than other methods in most datasets. The drop in its recall is more significant than that of its precision compared to the proposed methods. This method assumes that trainer's belief is fixed, therefore, it picks samples from a subset of the data based on the early users' annotations. However, some of the early annotations might be erroneous. It cannot repair its early decisions on determining the subset of the data for selecting examples. Thus, its models are biased toward the early (inaccurate) decisions, which deliver low recall. But, the learner that uses this method still takes advantage of the correctly labeled examples in the later stages of interaction to improve the precision of its models. The results of using other priors for learner and trainer and degrees of violations show a similar trend and are not reported due to the lack of space.

Appendix D. Related Work

Learning From Imperfect Annotators. There is a stream of papers on active learning in challenging settings where some (weak) users can return incorrect labels Zhang and Chaudhuri (2015); Shivaswamy and Joachims (2015), are allowed to relabel examples Yan et al. (2016), or even abstain from labeling. These approaches do not consider user learning during the annotation process. We believe combinations of these proposals and exploratory training should be explored in follow up work.
Inter-active Visual Labelling. Researchers have extended active learning for visual data analytics by allowing users to select what data item to label or to directly manipulate the current learned model Höferlin et al. (2012). By offering additional methods of leveraging users' expertise, this method can reduce the number of interactions in active learning Bernard et al. (2018b,c,a). As opposed to our setting, these methods assume the user knowledge about the target model is fixed.
Uncertain and Weak Labeling. In many domains, there might not be sufficiently many correctly labeled training data. In these settings, some weak prior beliefs about the training data might be available, e.g., using negative examples or beliefs about subsets of training examples. Authors in Rolf et al. (2022) leverage such set of beliefs to learn accurate labels for training data using Bayesian reasoning. Similarly, Varma and Ré (2018) assumes that correct (hard) labels are available for a small subset of training examples and aims at using them to learn heuristics to label the rest of examples. In our setting, trainers might also start with weak prior beliefs, but their beliefs evolves over time. To reduce the number of interactions, an interesting future work is to use the ideas in the aforementioned line of work to update the model of the learner and its sampling strategy after sufficiently many (early) interactions. One could use the model in Rolf et al. (2022) to investigate the degree by which the converged belief generalizes to the data items not presented in the interaction.