Towards Trustworthy Personalized Machine Learning Systems under User-System Interactions

Feng Lin

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2026

Reading Committee:

Shuai Huang, Chair

Cynthia Chen

Chaoyue Zhao

Prashanth Rajivan

Simon Shaolei Du

Program Authorized to Offer Degree:
Industrial & Systems Engineering

University of Washington

Abstract

Towards Trustworthy Personalized Machine Learning
Systems under User-System Interactions

Feng Lin

Chair of the Supervisory Committee:
Professor Shuai Huang
Department of Industrial & Systems Engineering

The revolutionary wave of Industry 5.0 heralds a new era of human-centered industrialization that moves beyond automation-centered production. Cutting-edge technologies, including artificial intelligence/machine learning (AI/ML) and big data, are increasingly expected to support human welfare and societal needs. In response to this emerging trend, the interests and experiences of real users have become central to the design of machine learning systems across various applications, such as healthcare, transportation, supply chains, etc. To build such ML systems, there are three critical factors to consider: 1) User Heterogeneity: Users differ in their needs, preferences, behaviors, and contexts. Therefore, ML systems should explicitly account for such heterogeneity and personalize services for individual users so as to improve service quality and user satisfaction. 2) System Trustworthiness: The real-world effectiveness of these personalized ML systems relies on sustained user engagement and feedback, which are grounded in user trust. These systems must therefore be trustworthy, protecting users' interests and mitigating potential risks arising from personalized services through characteristics such as robustness and fairness. 3) User-System Interactions: Instead of one-way delivery, ML systems should interact with users to collect their feedback and enhance their engagement under their behavioral uncertainty or environmental complexity. These factors highlight the system-level challenges central to building trustworthy

personalized ML systems that engage heterogeneous users.

To address these challenges, this dissertation contributes AI/ML methodologies that integrate these three aspects. **Chapter 2** focuses on the representation of user heterogeneity at scale and presents CrowdLLM, a framework that combines large language models (LLMs) and generative models to create high-fidelity digital populations for more cost-effective decision-making. **Chapter 3** and **Chapter 4** center on two key dimensions of trustworthiness: fairness and robustness. **Chapter 3** develops fair collaborative learning (FairCL), a framework that can incorporate a variety of fairness concepts to improve fairness amid personalization, along with a self-adaptive algorithm for effective implementation. **Chapter 4** further studies user-system interactions in reward systems and presents a robust learning method for modeling user choice behaviors under such interactions.

Taken together, this dissertation takes a step toward advancing trustworthiness and personalization in ML systems under user-system interactions. Its contributions lay a foundation for broader efforts to develop human-centric AI-enabled systems that can adapt to user disparities and support complex decision-making with real-world settings in the future.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Shuai Huang. Since the first day of my Ph.D. journey at the University of Washington, he has been unwavering in supporting, guiding, and mentoring me, not only in developing this dissertation and my research agenda, but also in my career development, professional growth, and many important decisions along the way. Without his insightfulness, patience, and encouragement, I could not have grown so confidently into an independent scholar.

I am also deeply grateful to my committee members—Dr. Cynthia Chen, Dr. Chaoyue Zhao, Dr. Prashanth Rajivan, and Dr. Simon Shaolei Du—for their dedicated support. Their valuable feedback, thoughtful questions, and diverse perspectives, have helped me strengthen the rigor of my work and sharpen my critical thinking.

My thanks also go to the staff of UW ISE, Jennifer Tsai, Kelly Fong, Sheila Prusa, Yuliya Shirokova and others, for their exceptional professionalism and administrative support. I can always turn to them whenever I need help navigating my academic life at ISE.

I could not have come this far without the support of my long-term collaborators. I especially thank Dr. Xiaoning Qian from Texas A&M University. I have been fortunate to work with him since the beginning of my Ph.D., and he has provided invaluable guidance as I entered the field of uncertainty quantification and Bayesian active learning. I am grateful to Dr. Li Zeng from City University of Hong Kong as well. Her thoughtful suggestions and support have contributed greatly to my professional growth. I also extend my gratitude to Dr. Chen Wang, my mentor during my internship at Mayo Clinic, and Dr. Ji Liu, my mentor during my time as a visiting researcher at Meta, for their patience and support in helping me connect my research with real-world industry practice.

I am sincerely grateful to Dr. Jianjun Shi and Dr. Jing Li at the Georgia Institute of Technology, as well as other senior members of the INFORMS QSR family. Their support and career advice have meant a great deal to me as a young scholar. I would also like to give special thanks to Dr. Ying Lin at the University of Houston, who has generously offered her guidance and encouragement throughout my PhD journey. I am also thankful to my colleagues for inspiring discussions and insights that enriched my research and Ph.D. experience. My wonderful lab mates, Congjing Zhang and Yuantao Wei, are incredibly reliable companions, always willing to listen, help and share their ups and downs with me.

I cannot imagine how lonely and difficult my Ph.D. journey would have been without the many wonderful friendships I built. My heartfelt thanks go to my best friend Tong Zhu, his husband Pengcheng Li, my steadfast ally Yankun Huang, and all the friends who have shared their companionship, encouragement, and care with me throughout this journey.

Finally, I owe my deepest thanks to my family for their endless support. To my parents, Yaping Lin and Xiaoxia Wang, thank you for your boundless love and unwavering support. To my chosen sister, Yiwei Fang and her husband Zixi Qi, thank you for your constant care and encouragement. To my dearest husband, Kehua Chen, thank you for your love, patience, and for always standing by me. It is because of you that I was finally able to reach this milestone and start the next chapter of life with gratitude and hope.

Looking back, when I came to Seattle alone in 2020, I was filled with excitement and hope, yet also with uncertainty and a sense of being lost. At that time, Seattle felt strange and unfamiliar to me. Over the years, however, it gradually became a place of growth, friendship, and belonging, and came to feel very much like home. Seattle witnessed my struggles, persistence, quiet moments of joy, and every meaningful encounter with the people I was fortunate to meet during my Ph.D. journey. As this journey comes to a close, I will carry these memories, lessons, and warmth of these years wherever I go. For all that this journey has given to me, I remain deeply grateful.

DEDICATION

To my beloved family.

To the love, courage, and all that we have built along the way.

To the brilliant adventure of life ahead of us.

"The machine does not isolate man from the great problems of nature, but plunges him more deeply into them."

— Antoine de Saint-Exupéry, *Wind, Sand and Stars*

Chapter 1

INTRODUCTION

1.1 Motivation

The rapid advancements of Artificial Intelligence/Machine learning (AI/ML) technologies and their deep integration into our daily life are bringing about a new era when the relationship between human and technology has shifted. Technologies are becoming more user-centered and emphasizing interactions with users, extending their roles beyond mere technological interventions onto roles in societal domains [2, 3]. While these emerging technologies have brought transformational changes to various systems, they also enforce more responsibilities and pose many methodological challenges [4, 5, 6]. Being user-centered not only requires the systems to cater for users ad hoc to improve their satisfaction, but also to build long-term connections and reach a win-win situation with users to achieve sustainability. For this short-term and long-term consistency, a good system should be able to take diversity and heterogeneity in the user population into account and continuously harvest their trust by providing high-quality and reliable service. By customizing the service delivered to different users according to their needs, the system will attract more users and increase their stickiness [7, 8]. The need for personalization is not rare in the real world [9], as people usually show different behaviors and have their own preferences. For example, in a video recommendation system, different users might have different interests in video content and probably react distinctly in terms of time spent on the videos, whether to like, whether to repost, etc. As a result, to increase user engagement, the system should personalize its policy of video content delivery for heterogeneous user cohorts. Another example is precision medicine. Different patients probably have different medical conditions and various lifestyles, which might lead to diverse trajectories of disease progression. Therefore, personalized risk

predictions and treatments are critical in preventive and diagnostic care for patients as they could maximally improve the unique conditions of each individual. This need for personalization has motivated extensive research in artificial intelligence and machine learning (AI/ML) [9], with widespread applications ranging from advertising [10, 11], to healthcare [12, 13, 14] and robotics [15, 16].

To realize a personalized ML system, it is essential to characterize user heterogeneity, which typically requires the analysis of individual-level user data. However, such data are often sparse, and collecting data from diverse users can be costly and time-consuming. Therefore, we need a scalable way to represent and simulate heterogeneous user populations. A natural question arises: Can we create high-fidelity digital populations that capture user heterogeneity and help with crowd-based decision-making while reducing the cost of human recruitment or data collection? The emergence of large language models (LLMs) has sparked much interest in creating LLM-based digital populations that can be applied to many applications such as social simulation, crowdsourcing, marketing, and recommendation systems. However, research has found that most of the existing works that rely solely on LLMs could not sufficiently capture the accuracy and diversity of a real human population. Therefore, we need a better way to handle such heterogeneity in creating digital populations while taking the advantage of the power of LLMs, which is the motivation behind Chapter 2.

Building personalized ML systems is not merely about simulation of user behaviors, but also enable individual decision-making through model personalization. If the user data is sufficient, model personalization can be realized by building models separately for different individual users with their own data. But in reality, users often have datasets of varying sizes and qualities. Insufficient or low-quality data will potentially introduce biases or noises, and mislead the system models. This motivates us to shift from one-size-fits-all solutions or fully individualized models to a better framework for personalization. On the other hand, as personalization indicates different treatments for different users, we are curious about how diverse they can be for heterogeneous users. The diversity in treatments might further cause inequality among users, which potentially makes them feel unhappy and reduces their

engagement. Consequently, it is important to make a trade-off between the diversity and equality, which motivates the discussion in Chapter 3.

Moreover, should we treat all the users equally regardless of their behaviors? In a large user population, there are probably some users who are malicious or robots [17, 18]. These users might inject misinformation or perform hazardous behaviors that potentially cause damages during their interactions to the system. Personalization for them is unnecessary and meaningless. On the contrary, we should identify these anomalies and rule them out to ensure the system can function normally and at least, be robust to them. These concerns of robustness, together with the aforementioned fairness and other concerns such as model interpretability and safety, etc., are parts of a trust between the real users and the system. Without the consideration of these issues in building a system, users can be unsatisfied and lose their trust in the system which is indispensable in building long-term user-system relationship. The discussion on the robustness in user-system interactions and related strategies are covered by Chapter 4 of this dissertation.

Together, this dissertation takes a step towards building human-centric AI-supported systems under a variety of system uncertainties and human related complexities. Figure 1.1 demonstrates an overview of the dissertation. More details about the chapters can be found in **Section 1.2**. The contributions of this dissertation are mainly methodological, and will have various applications in healthcare, medicine and transportation, etc. The approaches and methods used in this dissertation include, but are not limited to, machine learning, deep learning, optimization, uncertainty quantification, etc.

1.2 Outline

This dissertation is organized as follows. In the rest of introduction, we review the literature and describe the contributions of the dissertation in detail. The dissertation further explores three aspects towards building trustworthy personalized ML systems.

Chapter 2 starts the discussion with the key challenge of user heterogeneity and focuses on creating digital populations to simulate crowd-based decision-making by heterogeneous users,

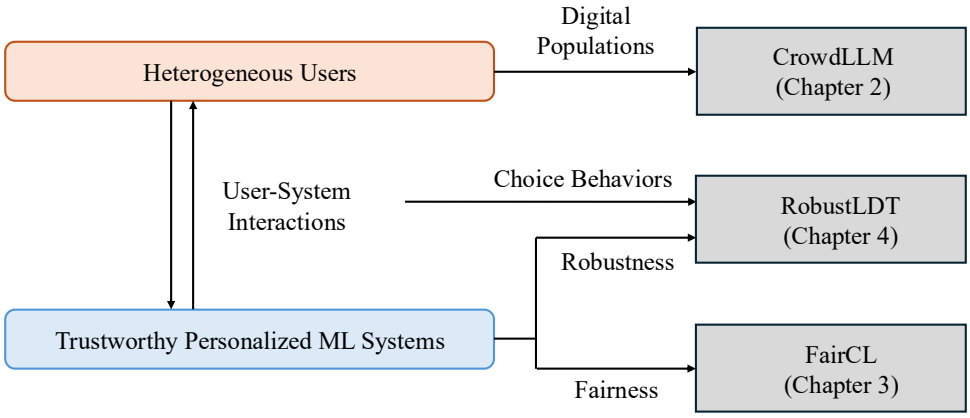


Fig. 1.1: A graphical overview of this dissertation.

which aims to reduce the cost of recruiting human participants and alleviate many concerns related to human subject study. To address the limitation of pure LLM-based simulation, we propose CrowdLLM that integrates pretrained LLMs and generative models to enhance the diversity and fidelity of the digital population. We conduct theoretical analysis of CrowdLLM regarding its great potential in creating cost-effective, sufficiently representative, scalable digital populations that can match the quality of a real crowd. Comprehensive experiments are also conducted across multiple domains (e.g., crowdsourcing, voting, user rating) and simulation studies which demonstrate that CrowdLLM achieves promising performance in both accuracy and distributional fidelity to human data.

Chapter 3 turns to the challenge of enhancing trustworthiness in personalized systems and moves to the discussion of fairness amid personalization which is a key dimension of trustworthiness for AI/ML. We describe the modeling dilemma in personalization and introduce the fairness issues amid personalization within the CL framework. We describe our proposed fair collaborative learning (FairCL) framework which explicitly add fairness control to CL and is versatile to incorporate a wide range of fairness metrics. We also introduce a self-adaptive learning algorithm for efficient learning of the personalized models. Experimental results show the superiority of our method in improving fairness over other personalization

methods.

Chapter 4 focuses on the robustness of personalized models. In this chapter, we limit our discussion to a new scenario for personalization, i.e., a reward system (e.g., travel demand management system, coupon system, etc.) where users are incentivized to share their likes or dislikes over offers given by the system. We present Latent Decision Threshold (LDT) for modeling user choice behaviors in reward systems and describe the challenges of bad actors and data corruption in such systems. We further introduce a robust learning method to mitigate the challenges and analyze its properties. Based on this method, we also develop a bad actor detection algorithm to screen out users that might harm the systems. The effectiveness of our method is evaluated on both simulated and real-world data.

Finally, Chapter 5 concludes this dissertation with a full discussion of the problems and techniques covered in the previous chapters. We also extend the discussion to some unsolved problems in personalized ML systems and wrap up our thoughts on future directions.

1.3 Literature Review

In this subsection, we review the existing research related to this work.

1.3.1 The Promises and Pitfalls of LLMs in Simulating Humans

LLMs have been used to simulate human behavior [19, 20], decision-making processes [21] and complex social interactions [22, 23]. For example, through a series of Trust Games grounded in behavioral economics and modeled with Belief-Desire-Intention reasoning, [24] show that GPT-4 agents exhibit strong behavioral alignment with humans in both actions and underlying rationales. Agent-based modeling with LLMs also shows great promise for large-scale social simulations. Frameworks such as AgentSociety [25] and SocioVerse [26] exemplify this potential, demonstrating simulations involving tens of thousands of LLM-driven agents or drawing upon millions of real users to inform agent behavior. These platforms aim to model complex societal dynamics, simulate millions of interactions, and study collective responses to events like policy changes or natural disasters. However, general-purpose LLMs can exhibit low accuracy on specific behavioral simulations. [19] shows that fine-tuned LLMs

[27] on behavioral data enriched with synthesized reasoning traces substantially improve the accuracy of action generation compared to training on actions alone. While many works demonstrated the promises of LLMs, there have also been many evidences that pointed out their pitfalls in generating human-grade data. For example, [28] use economic games to demonstrate that LLM behaviors are not consistent with humans and fine-tuned LLMs may only mimic specific patterns or contexts with reduced diversity even in simple scenarios. Beyond these inconsistencies, LLMs have also been observed to be associated with diminished output diversity [29, 30, 31].

To address the limited diversity and reliability in LLM-generated simulation outputs, [32] propose the LLM-as-a-Personalized-Judge framework and reveal that integrating verbal uncertainty estimation improves alignment with human judgments. [33] propose a multilingual prompting strategy to increase diversity by activating cultural knowledge embedded in model training data. Similarly, [34] introduce methods for evaluating and mitigating representational bias in LLM-driven outputs, ensuring that outputs better reflect a wide range of demographic and cultural perspectives. In addition, [35] emphasizes the importance of refining training datasets to reduce biases and improve both accuracy and fairness. Moreover, [36] highlight the role of hybrid human-LLM teaming to enhance model performance, demonstrating that human feedback can mitigate errors in complex simulations. These approaches aim to tackle issues of diversity and accuracy of LLMs, making them more reliable and representative for practical applications.

1.3.2 LLM-Based Digital Populations

In this subsection, we review existing works that create digital populations/synthetic crowds for applications such as voting, crowdsourcing, and product reviews. For example: (1) *Crowdsourcing*: Crowdsourcing [37] leverages the collective intelligence of workers who are usually non-experts to perform tasks such as labeling, classification, and data verification. The quality of crowdsourced data is often a challenge due to worker inconsistency, spammers, and labeling noise. Recruiting workers and ensuring response quality is time-consuming and

costly. LLM-based agents could circumvent many of these issues. Costabile et al. [38] suggest that an LLM-based crowd might outperform human crowds in fact-checking tasks by exhibiting less bias and higher consistency. Moskovskiy et al. [39] find that with techniques like activation patching, LLMs can generate parallel data with quality rivaling human-annotated corpora. However, Veselovsky et al. [40] suggest that only using LLM-based agents may be problematic in crowdsourcing scenarios considering their limitations in capturing the full range of human preferences and viewpoints. Wu et al. [41] investigate LLMs as workers in complex human-computational algorithms, observing variable success but highlighting potential for LLMs to handle sub-tasks within larger pipelines. To solve challenges in tasks requiring aligned and nuanced rewriting, Zeng et al. [42] propose hybrid aggregation strategies that combine LLM and crowd judgments for misinformation detection. These studies suggest LLM-based agents alone are insufficient, so some researchers explore different ways of human-LLM collaboration. E.g., Creator-Aggregator Multi-Stage [43], where LLMs and humans team up, aims to leverage mutual strengths by having humans come up with initial drafting and use LLMs, humans, and models to generate text answer aggregation. Li [44] shows that selective integration of LLM annotations can enhance overall annotation quality in both full and few-crowd settings. Tamura et al. [45] uses simulation-based approaches to understand optimal aggregation strategies in a human+AI crowd. (2) *Synthetic users in recommendation systems*: LLM-powered user simulators have become an important tool for recommendation systems by generating high-fidelity and interpretable synthetic interaction data that alleviates data sparsity and reduces the cost of online exploration. Recent advances take this idea in different directions: Agent4Rec [46] emphasizes population diversity by initializing agents with heterogeneous traits; RecAgent [47] prioritizes behavioral fidelity through cognitive components such as memory, reflection, and planning; SUBER [48] focuses on controllability and reproducibility for long-horizon evaluation, and [49] enhances transparency by explicitly modeling user preference logic and mitigating hallucination through an ensemble of logical and statistical components. Together, these simulators offer interpretable and adaptable user behavior models for evaluating recommendation policies, but

they also face shared limitations, including the high computational cost of cognitively rich agents and the challenge of balancing population diversity with stable, non-drifting within-agent preferences. (3) *Voting*: LLMs have been explored in electoral contexts, but their use raises concerns regarding consistency, fairness, and reliability in collective decision-making. Studies reveal issues across different elections: [50] observed biases and inconsistencies in LLM responses during the 2024 U.S. presidential election, while [51] reported failures in LLM-based predictions of the 2024 European Parliament elections, particularly in handling diverse national and linguistic contexts. Approaches such as fair voting aggregation have been proposed to mitigate these effects [52]. Despite these interventions, these observations highlight broader limitations of general-purpose LLMs in voting scenarios, particularly their limited capacity to capture human variability and their reliance on overly simplistic decision aggregation mechanisms. First, LLMs demonstrate a lack of diversity in synthetic outputs. [53] observed that LLM-generated data fails to replicate the variance seen in real human responses, with limited differentiation in persona-to-party mappings. Consistently, [54] showed that LLM outputs produce less diverse collective outcomes in simulated voting scenarios, and their data is used in our experiments, where our method improves both consistency and diversity. Second, LLM-based collective decision-making systems exhibit limitations in decision mechanisms, often relying on simplistic aggregation methods such as plurality or dictatorial voting, which constrains collective reasoning and robustness [55].

It is worth noting that, most of these LLM-based simulators emphasize fine-grained, per-individual behavior modeling in specific contexts. They are typically built on direct prompting or fine-tuning of LLMs, and often require substantially large data, costly agent architectures, or long-horizon interaction dynamics. As a result, there remains a gap in the literature for a lightweight, general-purpose framework for crowd-based decision making.

1.3.3 Generative Models

Generative modeling methods aim to learn the underlying data distribution to capture complex latent structures and variability, enabling models to generalize across diverse scenarios.

Over time, this goal has driven the development of several major paradigms. Variational Autoencoders (VAEs) [56] introduced a stable, likelihood-based framework in which data are encoded into a latent distribution and sampled through the reparameterization trick, enabling efficient conditional generation. Although VAEs may produce slightly smoothed outputs and fewer extreme samples due to the variational approximation, they remain tractable, stable to train, and easily conditioned on input variables. Generative Adversarial Networks (GANs) [57] enhance sample fidelity by training a generator against a discriminator, but this adversarial setup introduces instability, mode collapse, and high sensitivity to hyperparameters, making controlled diversity difficult. Diffusion models [58] achieve strong distributional coverage through iterative denoising, yet they require heavy computation, large datasets, and slow sampling, limiting their practicality in low-dimensional or conditional settings. Optimal transport–based models [59] and normalizing flows [60] provide exact likelihoods through invertible mappings but impose structural constraints that restrict flexibility in conditional scenarios. While each method addresses certain weaknesses, they also introduce trade-offs in stability, controllability, or computational cost. For generating structured latent variations, a VAE provides a stable and tractable probabilistic framework, making it the natural choice for the belief-generation component of CrowdLLM.

1.3.4 Fair machine learning

Fairness in machine learning has attracted tremendous attention from the research community [61, 62, 63]. Even though there is no consensus on the universal definition of fairness[64], researchers have proposed various metrics from different perspectives to quantify fairness in learning systems and develop specific algorithms to reduce biases. Given that this is a vast field, here, we will focus on the most relevant work that concerns the design of the learning formulations and develops fairness metrics or constraints.

One of the earliest and most widely used metrics is demographic parity (DP), also known as statistical parity (SP) [65, 66], which aims to ensure that the distribution of the model outcome remains the same regardless of sensitive attributes such as gender or race. Another

similar notion is disparate impact (DI) [67] that uses the probability ratio between protected and unprotected groups to measure fairness. However, these metrics can only support pairwise comparison and their effectiveness might be undermined when the sensitive attribute is not binary. To handle more general sensitive attributes, Jiang *et al.* further proposed generalized demographic parity (GDP), which extended the parity-based metric to incorporate continuous sensitive attributes while maintaining computational efficiency [68]. Different from parity-based metrics, equal opportunity [69] and equalized odds [70] were proposed to take not only predictions but also actual outcomes into account. Such group metrics more or less help with balancing between groups, but they may ignore protections for specific individuals in a group. In addition to these metrics, there are also other metrics that are not limited to comparison across groups. For example, Dwork *et al.* [71] presented individual fairness (IF) to advocate that similar individuals should receive similar treatment. However, this metric is also criticized for potentially leading to universal rejection and the difficult in measuring similarity [72]. As the fairness metrics might be controversial in their use, it is important to choose appropriate metrics in different contexts, which remains challenging.

With the emergence of many fairness metrics, researchers are also developing generic optimization formulations and computational strategies to incorporate these metrics into various models. For example, [73] developed an efficient optimization framework for fairness by minimizing the specific bounded group loss (BGL). To incorporate more general fairness metrics, from the perspective of non-convex optimization, [74] proposed to express fairness as a kind of "rate constraint" in "proxy-Lagrangian" formulation and provided a theoretically guaranteed procedure to solve it. This framework provides a general way to incorporate fairness as constraints, but suffers from dealing with nonlinearity in such constraints. [75] narrowed the problem down and proposed a flexible constraint-based framework enabling the design of fair margin-based classifiers. This framework uses covariance between sensitive and non-sensitive attributes as a tractable proxy for unfairness and aims to limit unfairness with small performance cost. However, it might result in poor performance when the data is unbalanced. While most of these studies focus on classification problems, there are also works

considering regression problems, e.g., [76, 77], which is beyond the scope of this dissertation.

1.3.5 Model personalization

Models that can be used for personalization include the classic mixed effects model [78, 79] and more modern ones such as transfer learning [80, 81] or multi-task learning [82, 83]. Transfer learning is a learning paradigm that improves learning in a target domain by transferring knowledge from a source domain [84] while multi-task learning aims to leverage information shared by multiple related tasks to help with learning for all the tasks [82]. Despite attaching different degrees of importance to various tasks of interest, both of them are similar in the strategies adopted for modeling [80]. Within the general umbrella of transfer learning, collaborative learning (CL) was developed [85, 86, 87]. It features a concept known as canonical models, which distinguishes itself from other transfer learning or multi-task learning models and employs a probabilistic relationship to characterize individual models.

Another line of work that is related to model personalization is distributed learning [88, 89], where data/computing resources are allocated to different clients. As a distributed learning approach which emphasizes data privacy protection, federated learning (FL) has become increasingly popular in recent years [90, 91]. To compensate for its poor performance caused by data heterogeneity, personalization techniques have started to be used in federated learning [92, 93, 94]. At first sight, this line of work is somewhat similar to CL. However, it is worth noting that there are several differences between them: 1) CL aims to model individuals by utilizing population-level information while personalization in FL strives for better FL on heterogeneous data. 2) CL allows the sharing of data together with model information among individuals while the privacy requirement in FL prevents data from being shared among clients and only allows sharing the knowledge of model updates. 3) CL assumes the individuals share a latent canonical structure while FL usually defaults to independence among clients. 4) CL emphasizes model personalization through centralized training but with no explicit central server while personalized FL allows training personalized models in a decentralized way through information sharing between an explicit central server

and clients. 5) CL is useful when individuals have different amount of data while traditional FL sometimes relies on sufficient data from its distributed clients to reach a minimum model quality [95, 96]. It should be noted that CL and FL are not competitive with each other, but instead, they solve different questions. And CL can be compatible with FL. Another approach worth mentioning is the clustered federated learning (CFL) [97, 98]. It addresses heterogeneity by explicitly identifying the clusters of the users and performing FL to learn separate models for each cluster instead of one global model. While some works use centralized clustering algorithms such as K-means [97, 99], hierarchical clustering [100, 101], etc., others use decentralized algorithms [102, 103] to recognize the cluster identities. In addition to hard clustering, [104, 105, 106] propose to use soft clustering strategies. CFL methods have provided a promising way to handle user heterogeneity through a clustering structure, but their performance depends on the ability to find good clusters. In contrast to CFL, CL doesn't need to cluster the users but encodes the sharing characteristics among the users through a latent canonical structure that is not necessarily a clustering structure, which also allows simple and efficient personalization through one unified optimization framework.

Finally, different from model personalization, a remotely related line of work is the design of personalization strategies. For example, personalized recommendation [107, 108, 109] is increasingly pervasive to handle user heterogeneity [9]. Such personalization is often conducted via collaborative filtering or content-based filtering. Collaborative filtering makes recommendations to an individual user based on the preference of other similar users, while content-based filtering recommends similar items to the user without taking other users' information into consideration. These strategies are tied to many specific application domains such as healthcare [110, 111], e-commerce [112, 113], education [114, 115], etc. They have prospered rapidly in recent years with the development of deep learning techniques [116, 117].

1.3.6 Fairness-aware federated learning

There have been works in federated learning (FL) focusing on ensuring fairness in different stages of FL, building on various fairness metrics proposed in literature [118]. Some studies

aim to enforce fairness constraints on client selection [119]. For example, [120] proposed to incorporate a fairness constraint in Lyapunov optimization to guarantee the participation rate of clients, while [121] and [122] used bandit-based algorithms to encourage a fairer selection. While these methods ignored real-time client contribution, it was further addressed in [123] by combining a reputation table with fairness constraints in selection. Other work promoted the selection of underrepresented clients through personalization of training procedures, such as [124, 125]. Nevertheless, as the biases in training may not just come from client selection, more studies have been concerned with fairness in the model optimization process [119]. [83] developed Ditto, a multi-task personalized FL framework to promote fairness in FL through regularization on the discrepancy between personalized models and the global model in local training. Similarly, [126] also introduced regularization terms in their GIFAIR-FL framework to ensure fairness. Instead of penalizing the models, they regularized between the local loss functions, and showed their formulation could be equivalently written as loss re-weighting. Re-weighting has also been applied in other recent work. Extending the idea of agnostic federated learning (AFL)[127] which optimized their loss over an unknown mixture of the client data distributions, [128] proposed AgnosticFair that re-weighted an agnostic loss function with kernel function parameterization and constructed an agnostic fairness constraint from demographic parity. However, its reliance on prior knowledge limits its applications [118]. [129] proposed q-Fair Federated Learning (q-FFL) where different clients are weighted differently in the aggregated loss through a power parameter q. But its effectiveness will be impacted by the estimation of a Lipschitz constant and the setting of q. When q is large, it may also suffer numerical issues in computation. In addition to model optimization, there are also studies aiming to improve fairness in the stage of incentive distribution, e.g., [130, 131, 132] took fairness into consideration through the incentive mechanism design.

1.3.7 User choice behavior modeling

Discrete choice models based on random utility maximization (RUM) have been widely applied in many areas such as transportation [133], emergence evacuation [134], and healthcare

[135], mainly for population-wide policy making and resource allocation purposes. Some recent papers used machine learning techniques to analyze and predict individual user behaviors. [136] presented an analysis of user behavior by a skip-or-stay behavior classifier. [137] discussed a car ownership problem and showed that machine learning models can perform better than discrete choice models in predicting users' demands. There are other lines of works that model user behaviors from cognitive and human factor perspectives. For example, [138] proposed an integrative model for choice behavior based on cognitive heuristics. Many of these models rely on specialized testing protocols and experimental tasks such as the Iowa Gambling Task (IGT) and the Soochow Gambling Task (SGT). On the other hand, inverse modeling methods have recently emerged as a powerful tool for choice behavior modeling [139, 140]. In the context of human computer interaction (HCI), [141] showcased the inverse modeling of users' interactive behaviors with approximate Bayesian computation (ABC) inference under the assumption of computational rationality. For other applications such as travel behavior modeling, inverse discrete choice models have also been explored to enrich data for capturing choice behaviors [142]. Taking advantage of deep neural networks, [143] proposed an inverse reinforcement learning (IRL) framework for modeling link-based route choices. There is also considerable interest in recommendation system research for the modeling of user behaviors. By learning from their purchase history, the recommendation systems aim to understand their preferences so as to recommend new products to encourage purchases, such as collaborative filtering [144] and deep learning techniques [145].

1.3.8 Robust learning

A traditional robust learning strategy is to borrow concepts from robust statistics such as the M-estimates, L-estimates to achieve robust estimation of the model parameters [146]. Some others adopt ideas from robust optimization and formulate min-max optimization problems [147, 148]. Besides these two general strategies, [149] proposed to learn a robust regression model with a bounded loss function by clipping. Regularization is also a widely used tool for robust learning. [150] showed that a regression model can be recovered via

simple ℓ_1 minimization. [151] further proposed an effective approach for robust regression through low-rank representations. Our work follows the same line as some studies on Robust Least Squares Regression (RLSR) that aim to achieve robustness by sample selection, such as the robust thresholding regression [152], thresholding operator-based robust regression [153], and consistent robust regression [154] etc. To select appropriate samples, they used an iterative hard-thresholding technique which had been widely used in compressed sensing and sparse recovery [155, 156]. While these works have been developed for regression problems, similar strategies could be used for robust classification. For instance, the strategies of robust optimization that consider the worst-case application and the min-max framework has been adopted in [157, 158], etc. Some other notable related works include those using L_1 regularization [159] or Huber loss [160] as their robust learning strategies, those using regularization on corrupted features [161, 162], or some other works as [163] when only class labels are noisy. [164] focused on robust logistic regression model and learned the model by solving a linear programming problem. [165] discussed an interesting setting where data used for learning comes from multiple untrusted sources.

1.4 Bibliographic Remarks

This dissertation includes the research in the following publications:

- Lin, Feng, et al. "CrowdLLM: Building LLM-Based Digital Populations Augmented with Generative Models." INFORMS Journal on Data Science (2026).
- Lin, Feng, et al. "Fair Collaborative Learning (FairCL): A method to improve fairness amid personalization." INFORMS Journal on Data Science 4.1 (2025): 67-84.
- Lin, Feng, et al. "Modeling user choice behavior under data corruption: Robust learning of the latent decision threshold model." IISE Transactions 56.12 (2024): 1307-1320.

Chapter 2

CROWDLLM: CREATE A DIGITAL POPULATION WITH LARGE LANGUAGE MODELS AND GENERATIVE AI

2.1 Introduction

Recent years have witnessed the immense potential of large language models (LLMs) in performing human tasks [166, 167] and human behavior simulation [168, 169]. This capacity of LLMs has sparked much interest in creating virtual human-like decision-making agents that can be used in many applications such as social simulation [170, 171], behavioral studies [172, 173], crowdsourcing [174, 175, 39], marketing [176, 177], recommendation [178, 179], etc. A common theme of these applications is that they all involve a large group of human participants so as to solicit their decision-making powers to provide solutions to a task, and then, aggregate their solutions to solve the task [180]. Apparently, an implicit assumption made in these "human-intensive" operations is that the system could not bet on one single participant to solve the problem. Instead, it relies on the wisdom of the crowd, which by definition means a diverse collection of individuals who would provide different responses on the same problem. There are many reasons: sometimes it is because there is no ground truth (like in voting); sometimes it is because the quality of the crowd is not guaranteed (like in crowdsourcing); and sometimes it is simply because the goal is to solicit input from the diverse population (like collecting user ratings for products in order to develop accurate recommendation systems). The growing interest in using LLMs in these applications is motivated by reasons that vary across the applications, but one common reason is that in many of these applications the involvement of real humans may raise many concerns about privacy [181], confidentiality [182], quality [183], transparency [184], etc. These concerns also extend to other fields, such as bias in social science [185] and privacy risks in recommendation

systems [186]. Beyond these legal, ethical, and cost-related challenges, recruiting workers itself presents significant complexities. The LLM-based synthetic crowd can circumvent these challenges and therefore inspire the many recent aforementioned developments. As articulated in [187], LLMs should always be used as a concept testing tool for pilot and exploratory studies before we recruit real humans.

Generally, an LLM-based model can be constructed based on a pretrained LLM, such as OpenAI ChatGPT [188], Meta Llama [189], Google Gemini [190, 191], Deepseek [192], etc. It is followed by supervised fine-tuning (SFT) on a specific task [193, 194], possibly combined with human preference alignment through learning methods such as reinforcement learning from human feedback (RLHF) [195, 196, 197], to achieve better performance in completing the given task. Although such a pipeline has been widely adopted, the underlying drawbacks are not negligible. While SFT is much cheaper than pretraining [198], it can still be cost-prohibitive, which further demands parameter-efficient techniques to reduce the cost [199, 194]. Meanwhile, SFT or RLHF can be largely impacted by the quality and the curation of the task-specific data used for tuning [200, 201, 202]. Thus, building a high-achieving LLM-based model to perform human tasks remains a challenge, particularly when there is a lack of high-quality data and computational resources.

Noting these issues, many endeavors have been devoted to the improvement of the model tuning [203, 204] and prompt designing [205, 206, 207] so as to craft well-tuned task-specific LLMs. However, LLMs are pure "black box" models whose controllability (i.e., of their behaviors and output) is known to be a challenge. While instruction prompts play a significant role in inducing LLMs to show desired behaviors, it is usually difficult to find a universal design of the prompts for various tasks. It is also beyond our reach to know whether LLMs really understand the prompts and generate the outputs causally based on the instructions. Additionally, LLMs usually generate outputs with little diversity [208, 209, 29], which is actually one key challenge in the development of the envisioned digital populations that we are interested here. It can be seen that many existing efforts are devoted to the framework of LLM which relies on large-scale high-quality data and substantial computational

resources, whereas in many aforementioned applications, sometimes a lightweight solution of the LLM-based digital population is sufficient for the task. After all, in these applications, such as crowdsourcing a data labeling task or surveying a potential market for a new product, the human participants need not be experts; they may perform poorly on individual tasks, yet through effective aggregation of their inputs, they can collectively achieve satisfactory results. Still, the main challenges for developing such an LLM-based digital population are, as pointed out in [187] and many other recent efforts, that these LLM-based models usually exhibit a lack of diversity and undetected bias, and inaccuracies due to excessively user-pleasing outputs.

Therefore, targeting the applications where a lightweight solution of the LLM-based digital population is sufficient, we pursue a strategy that is different from the existing efforts that aim to solve the problem within the LLM framework itself. Rather, our approach is to augment LLM with a generative machine learning model that can provide the diversity it needs, mitigate the bias it implicitly has, and improve its accuracy. We develop a principled design of a computational pipeline that is lightweight enough to be cost-effective but also sufficiently accurate and robust to guide the generation and aggregation of a diverse pool of LLM-based virtual participants to match the diversity and accuracy of real-world operations.

Our main contributions include: (1) we propose CrowdLLM to emulate decision-making diversity and distributional fidelity observed in many real-world operations in crowdsourcing, voting, and product reviews; (2) CrowdLLM is built on a rigorous probabilistic framework that integrates the best of the two worlds, the LLM and the generative ML models; (3) we conduct theoretical analysis of CrowdLLM regarding its great potential in creating cost-effective, sufficiently representative, scalable digital populations that can match the quality of real populations; and (4) we conduct comprehensive experiments across multiple domains (e.g., crowdsourcing, voting, user rating) and simulation studies which demonstrate that CrowdLLM achieves promising performance in both accuracy and distributional fidelity to human data.

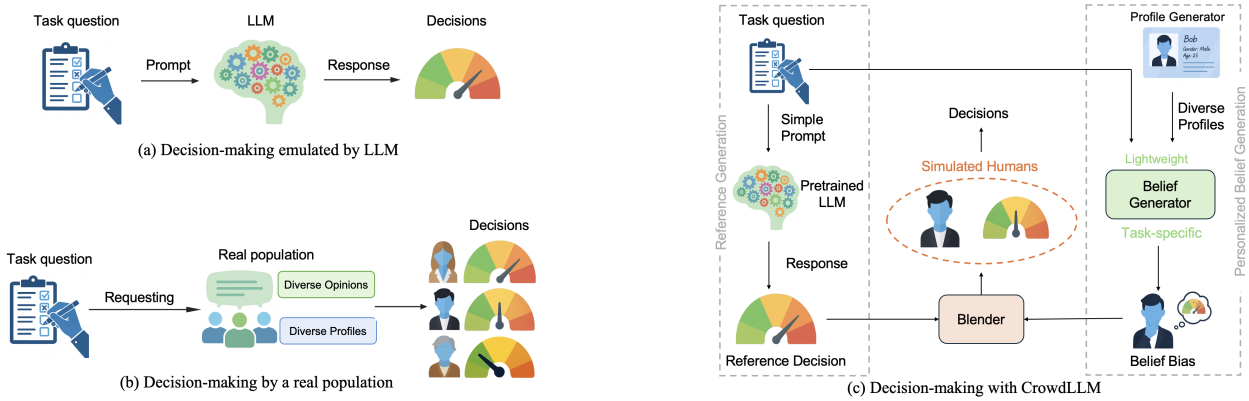


Fig. 2.1: A comparison of different decision-making workflows. (a) LLM: Decisions are purely made by LLM through the input of prompts. (b) Real population: Diverse decisions are made by a population of humans with diverse profiles. (c) CrowdLLM: Diverse decisions are made by simulated humans. Each simulated human's decision is a blend of a reference decision generated by a pretrained LLM and the personal belief bias generated by a belief generator. The simulated humans are sampled probabilistically by a profile generator.

2.2 CrowdLLM

Our target applications involve a group of human participants to solicit their decision-making powers to provide solutions to a task, and then aggregate their solutions to solve tasks. Before formally presenting our framework, let's first provide an analytic characterization of these applications. For a specific decision-making task, we consider a set of T problems $\mathcal{T} = \{1, 2, \cdots, T\}$. Each problem $t (t \in \mathcal{T})$ is associated with a description $\boldsymbol{x}_t \in \mathcal{X}$, where $\mathcal{X}$ is the problem space. Given $\boldsymbol{x}_t$, one needs to give their individual response. For example, in a choice-making scenario, they need to make a choice y_t from the set of M_t alternatives $\mathcal{Y}_t = \{1, 2, \cdots, M_t\}$. The ultimate goal of decision-making is to find an optimal rule $\psi : \mathcal{X} \to \mathcal{Y}$ to give the decision $\hat{y} = \psi(\boldsymbol{x}_t)$. Suppose we have N participants, and each is with a profile vector $\boldsymbol{v}_i (i = 1, \cdots, N)$, i.e., which includes their demographics or user characteristics. Each problem will be assigned to all the participants, but they can choose whether to perform

the task or not. Thus, for each problem t, we will only collect responses from a set of N_t participants, denoted by $Y_t = \left\{ y_{t,i_n} \in \mathcal{Y}_t | n = 1, \cdots, N_t \right\}$. With an aggregation function $h(\bigcup_{i=1}^{N} \{y_{t,i}\})$, the final decision for the problem can be represented by $\hat{y}_t = h(Y_t)$. As a result, the decision-making rule is de facto an ensemble of personalized decision-making rules. Both personalization and aggregation are important in our target applications. While aggregation synthesizes the collective wisdom, personalization emphasizes the diversity of participants in both their profiles and opinions, as illustrated in Figure 2.1(b). The overall framework of CrowdLLM is shown in Figure 2.1(c). Different from a pure LLM-based model shown in Figure 2.1(a), in CrowdLLM, the LLM-emulated participants are augmented with a generative model to mimic the task-specific behaviors of real human participants. Note that rather than building numerous personalized LLMs tailored for each individual (pure LLM agents), CrowdLLM allows all these virtual individuals to share one single pretrained LLM as their engine, which is a more cost-effective solution. A full description of CrowdLLM is shown in the box below, and more details are given in the rest of this section.

2.2.1 Details of Each Component in CrowdLLM

Reference generation. Recall that for any given problem/task $\boldsymbol{x}_t$, any decision CrowdLLM generates combines two inputs, i.e., as illustrated in Figure 2.1(c), the reference decision and the personal belief. To produce the reference decision for problem t, we leverage LLM, in particular, a pretrained LLM $\mathcal{M}$, since it is computationally cheaper and imposes fewer requirements on specialized hardware compared to fine-tuning [210]. For a single LLM-emulated participant, with $\mathcal{P} = f(\boldsymbol{x}_t, \mathcal{R}_t, \mathcal{C}_t)$ as context, i.e., recall that each problem t has its description $\boldsymbol{x}_t$, requirements $\mathcal{R}_t$ and context $\mathcal{C}_t$ (see an example of $\mathcal{R}_t, \mathcal{C}_t$ in Appendix), we prompt $\mathcal{M}$ to generate several decisions, which we call reference decisions. This can be viewed as sampling from a reference distribution π_M over $\mathcal{Y}$, i.e., $y_{\text{ref}} \sim \pi_{\mathcal{M}}(y|\mathcal{P})$. To ensure the reliability of the reference decision, we perform K times of generation, which yields a set of decisions $\{y'_1, \cdots, y'_K\}$. In summary, the reference decision can be expressed as an

aggregated decision:

$$y_{\text{ref}} = h_{\mathcal{M}}(y'_1, \cdots, y'_K),$$
$$y'_k \sim \pi_{\mathcal{M}}(y|\mathcal{P}), \quad k = 1, \cdots, K,$$

where $h_{\mathcal{M}}(\cdot)$ is an aggregation function, e.g., mean or majority voting. Though as a good common sense respondent, it is known that LLM-based agents often fail to generate differentiated decisions but instead follow the same common sense [40, 34, 211], even when we vary the ways of prompting (multi-persona prompting) and temperature settings. Our experiments in Section 4 also show that the LLM-emulated participants lack diversity compared with real humans' decisions.

Belief generation. To ensure the LLM-emulated participants can make diverse decisions as humans, we introduce a belief generator $G_{\text{belief}}(\cdot)$ to generate personalized belief biases. The generator can be implemented as a lightweight generative network that can adapt to a specific task. It takes both the participant's profile $\boldsymbol{v}_i$ and the problem description $\boldsymbol{x}_t$ as the input, and encodes them into a belief bias. The generation of the belief bias follows an inference model:

$$\begin{aligned} \boldsymbol{\delta}_{i,t} &\sim p(\boldsymbol{\delta}|g_x(\boldsymbol{x}_t), g_z(\boldsymbol{v}_i)) \\ &= \mathcal{N}\Big(\boldsymbol{\mu}\Big(g_x(\boldsymbol{x}_t), g_z(\boldsymbol{v}_i)\Big), \boldsymbol{\Sigma}\Big(g_x(\boldsymbol{x}_t), g_z(\boldsymbol{v}_i)\Big)\Big), \end{aligned} \tag{2.1}$$

where $g_x(\cdot)$ and $g_z(\cdot)$ are embedding functions parameterized with $\boldsymbol{\beta}$. When fixing the participants' profiles $\boldsymbol{v}_i$ as the context, this naturally leads to a variational autoencoder (VAE) conditioning on the profiles. But it should be also noted that if the profile generator is not frozen and $\boldsymbol{v}_i$ can vary with the noise $\boldsymbol{\varepsilon}_i$ that generates the profiles, $p(\boldsymbol{\delta}|g_x(\boldsymbol{x}_t), g_z(\boldsymbol{v}_i))$ is not necessarily Gaussian after marginalization and thus can result in a semi-implicit variational autoencoder which is able to accommodate non-Gaussian distributions through a hierarchy of stochastic layers [212]. To simplify the problem, we directly go with the VAE structure without considering such hierarchical inference. The reconstruction of the problem description is performed by a decoder $D(\cdot)$ through $\boldsymbol{x}_t = D(\boldsymbol{\delta}_{i,t}, g_z(\boldsymbol{v}_i))$. And the generated belief

CrowdLLM: A Synthetic Crowd of Human Participants

Input: Candidate pool $\mathcal{S}$; Recruitment budget N; Task-specific problem set $\mathcal{T} = \{1, \cdots, T\}$, each problem t with its description $\boldsymbol{x}_t$, requirements $\mathcal{R}_t$ and context $\mathcal{C}_t$; Frozen LLM $\mathcal{M}$.

1. **Virtual Participant Recruitment:** Produce a set of N participants with qualified profiles $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N$ from the candidate pool for problem t. For each problem t, assign the problem to the participants and ask them to make decisions.
2. **Reference Generation:** Instruct the LLM $\mathcal{M}$ with problem-specific prompt $\mathcal{P} = f(\boldsymbol{x}_t, \mathcal{R}_t, \mathcal{C}_t)$ to generate reference decisions $y_{\text{ref}} \sim \pi_{\mathcal{M}}(y|\mathcal{P})$ for the problem t.
3. **Belief Generation:** For the i-th participant, if their participation status $\varphi_{t,i} = 1$, generate their belief bias over the problem as $\boldsymbol{\delta}_{i,t} = G_{\text{belief}}(\boldsymbol{x}_t, \boldsymbol{v}_i)$.
4. **Personalized Decision-Making:** For the i-th participant, the personalized decision is made by a blending of the reference decisions and personalized belief bias which follows a probabilistic model $\hat{y}_{i,t} \sim \pi(y|B_\sigma(y_{\text{ref}}, \boldsymbol{\delta}_{i,t}), \boldsymbol{x}_t, \boldsymbol{v}_i)$ where $B_\sigma(y_{\text{ref}}, \boldsymbol{\delta}_{i,t})$ is the blender.
5. **Decision Aggregation:** Depending on the task, we can aggregate the decisions for any problem t by a function h through $y = h(Y_t)$, where $Y_t = \{\hat{y}_{t,i}|\varphi_{t,i} = 1\}$.

bias $\boldsymbol{\delta}_{i,t}$ is then fed into the blender (shown in Figure 2.1(c)) to produce the final decision from this virtual participant.

Personalized decision-making. To finalize the decision of a single participant, we need to blend the reference decision generated by LLM with the personal belief bias that is sampled from (2.1) as a latent vector. Since (2.1) is a probabilistic model, to offset the impact of its randomness, we generate the final decision of the participant as an expected

decision:

$$\hat{y}_{t,i} = \mathbb{E}_{\boldsymbol{\delta}_{i,t}\sim p(\boldsymbol{\delta})}\Big[B_\sigma(y_{\text{ref}}, \boldsymbol{\delta}_{i,t})\Big] \approx \frac{1}{J}\sum_{j=1}^{J} B_\sigma(y_{\text{ref}}, \boldsymbol{\delta}_{i,t}^{(j)}).$$

In practice, we can generate the personal belief bias J times and approximate the expectation by the sample average. Here, $B_\sigma(\cdot)$ is the blender parameterized by σ. In our framework, the blender follows

$$\tilde{y}_{t,i} = B_\sigma(y_{\text{ref}}, \boldsymbol{\delta}_{i,t}) \sim \mathcal{F}(y_{\text{ref}} + \boldsymbol{\delta_i}, \sigma^2), \tag{2.2}$$

where $\mathcal{F}$ is a preset distribution depending on the task, and the variance σ^2 reflects the noise level. Here, $\mathcal{F}$ is chosen flexibly to accommodate different task structures. For example, if the decision to make is a continuous variable, $\mathcal{F}$ can be a normal distribution. For discrete choices, $\mathcal{F}$ can be parametrized by compounding a latent continuous score distribution with a discretization mapping, i.e., $\mathcal{F}$ can be a composition of normal distribution that treats $y_{\text{ref}} + \boldsymbol{\delta_i}$ as a latent score and a task-specific discretization operator. For example, if the decision to make is binary, $\mathcal{F}$ can be a logit-normal distribution.

Crowd-level decision aggregation. In the final step, for each problem t, the participants' decisions are aggregated through an aggregation function $h : \mathcal{Y}^N \to \mathcal{Y}$. Specifically, the aggregated response for problem t can be written as

$$y = h(Y_t), \text{where } Y_t = \{\hat{y}_{t,i} | \varphi_{t,i} = 1\}.$$

where $\varphi_{t,i} = 1$ means participant i responds to problem t. Various aggregation functions can be used, such as mean score, majority voting, Dawid-Skene model, etc. When the problems do not have a ground truth solution, the consensus or the decision distribution of human workers can be the gold standard to evaluate the performance of CrowdLLM.

Virtual participant recruitment. Last but not least, recruitment of LLM-emulated participants is realized through a random profile generator $G_u(\cdot)$, i.e., this profile generator

should be responsible for generating the information of a participant and selecting participants from a pool of qualified profiles denoted by $\mathcal{S}$. $\mathcal{S}$ can be built based on task-specific prior knowledge (see an example in Figure 2.3). Formally, the i-th participant's profile is:

$$\boldsymbol{v}_i = G_u(\mathcal{S}, \boldsymbol{\varepsilon}_i; \boldsymbol{\theta}), \tag{2.3}$$

where $\boldsymbol{\varepsilon}_i \sim q(\boldsymbol{\varepsilon})$ is random noise encouraging profile diversity and $\boldsymbol{\theta}$ is the generator's parameter. Once we obtain the profile of a participant, we can simulate one's decision-making behaviors through the generation components of CrowdLLM (i.e., from reference generation to decision aggregation). We also consider that in reality not all participants participate in all problems. We assume the participation of the i-th participant on problem t, $\varphi_{t,i}$, satisfies a Bernoulli distribution $\varphi_{i,t} \sim Bernoulli(p_{i,t})$, where $p_{i,t}$ is the probability of participation which can be defined by prior knowledge. We treat participation as a reflection of prior knowledge to enable the digital population to better reflect the real-world scenarios where individuals selectively engage in tasks based on factors such as availability, expertise, or profile characteristics. Thus, we would recommend deriving these probabilities from observed human participation patterns in the historical data. Only when no prior information is available, these probabilities can be chosen uniformly.

2.2.2 Model Training

Training of CrowdLLM will only require a small set of real human data, since the LLM-emulated participants are built on a frozen pre-trained LLM and only the generators and the blender need to be trained. The training data includes multiple problems/instances and the decisions of a set of real human participants for each problem/instance. To train CrowdLLM, the loss functions are:

$$\begin{aligned}
\mathcal{L}_1 =& \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i}\varphi_{i,t}\Big\{\mathbb{KL}\Big(\mathbb{E}_{\Omega\sim q_\phi(\Omega|\boldsymbol{x}_t,\boldsymbol{v}_i)}[q_{\boldsymbol{\beta}}(\boldsymbol{\delta}_{i,t}|\Omega)]\Big\|p(\boldsymbol{\delta}_{i,t})\Big) \\
& - \mathbb{E}_{\Omega\sim q_\phi(\Omega|\boldsymbol{x}_t,\boldsymbol{v}_i)}\Big[\mathbb{E}_{\boldsymbol{\delta}_{i,t}\sim q_{\boldsymbol{\beta}}(\boldsymbol{\delta}_{i,t}|\Omega)}\Big[\log p(\boldsymbol{x}_t|\boldsymbol{\delta}_{i,t},\boldsymbol{v}_i)\Big]\Big]\Big\}, \\
\mathcal{L}_2 =& \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T_i}\sum_{t=1}^{T_i}\varphi_{i,t}\ell(\hat{y}_{i,t}, y_{i,t}),
\end{aligned} \tag{2.4}$$

where $T_i = \sum_{t=1}^{T} \varphi_{i,t}$, $q_\phi(\Omega|\boldsymbol{x}_i, \boldsymbol{v}_i)$ is an implicit prior distribution, and $q_{\boldsymbol{\beta}}$ is the variational distribution of the belief bias conditioned on the implicit parameter Ω. Here, $\mathcal{L}_1$ follows the semi-implicit VAE style [212] and ensures the model sufficiently represents the personal belief of the human participants, while $\mathcal{L}_2$ ensures the final accuracy of CrowdLLM, i.e., the final decision made by a virtual human participant should be close to its real human counterpart's. The specific form of $\ell(\cdot, \cdot)$ in $\mathcal{L}_2$ depends on the task scenario. For regression-type continuous or ordinal judgment problems, the squared loss $\ell(\hat{y}_{i,t}, y_{i,t}) = ||\hat{y}_{i,t} - y_{i,t}||_2^2$ is a common choice; For classification-type choice-making problems, the cross-entropy loss which reduces to $\ell(\hat{y}_{i,t}, y_{i,t}) = -\log p(\hat{y}_{i,t} = y_{i,t})$ is widely adopted. In practice, a simpler alternative is to reformulate the classification problems as continuous ones by introducing latent scores with the blender and optimizing the regression-type squared loss. The overall loss function is $\mathcal{L} = \mathcal{L}_1 + \lambda \mathcal{L}_2$, where $\lambda > 0$ is a regularizer. By minimizing the loss function $\mathcal{L}$, we can optimize the parameters of CrowdLLM.

2.3 Theoretical Analysis

In this section, we perform theoretical analysis to further reveal the underlying mechanisms of CrowdLLM about why it can create realistic digital populations. Specifically, we ask the following questions: (Q1) Is CrowdLLM able to generate a target population with envisioned profile characteristics? (Q2) How does the diversity of the digital population generated by CrowdLLM affect the decision-making performance (e.g., accuracy)? And (Q3) How is the decision-making performance impacted by the quality of the LLM backbone and the generative models in CrowdLLM? All the proofs are in the Appendix.

To answer Q1, we can readily extend the theoretical results of generative models in the literature [213, 214]. Specifically, the following theorem shows that it is possible to generate a diverse population of a target profile distribution $\mathcal{T}$ through the profile generator in CrowdLLM:

Theorem 2.3.1. *Suppose the profiles are d-dimensional bounded vectors following a target mixed-type distribution $\mathcal{T}$. Consider ρ as an easy-to-sample distribution taken to be uniform*

on $(0,1)^{d+1}$. *For any* $\varepsilon \in (0,1)$, *there exists a profile generator* G *building on a generative model that satisfies*

$$W_1(G_\sharp \rho, \mathcal{T}) < (1 + \sqrt{\frac{2}{\pi}}\eta d)\varepsilon.$$

Here, W_1 *is the Wasserstein-1 distance,* $G_\sharp \rho$ *is the pushforward of* ρ *by* G *which represents the resulting distribution transferred from* ρ *to the generated profile space, and* η *is a constant.*

Theorem 1 indicates that we can build a profile generator to generate meaningful profiles of a target population. Rather than being tied to any specific architecture, the profile generator can be implemented flexibly, ranging from learned generative models to even simpler sampling-based schemes. When profile pools are complex and the induced search spaces are large, the generator can adopt common generative model classes (e.g., GANs, VAEs, flow-based models, etc.) for constrained generation. However, for smaller and finite pools, a simpler sampling-based generator (e.g., multinomial sampler) is sufficient. However, generating a diverse profile of the digital population doesn't guarantee CrowdLLM's good performance, as the virtual human participants, despite having a diverse profile, may still give similar responses on the same task. Thereby in CrowdLLM we further have the belief generation component to ensure that human participants can generate different responses as they have different profiles.

Now we answer Q2. Consider a task that is characterized by the problem description $\boldsymbol{x}$. The Bayes-optimal response on this task is the conditional mean response given by the target human population as $y^* = \mathbb{E}_{\boldsymbol{v}\sim\mathcal{T}}\mathbb{E}_{y\sim\mathcal{Y}|\boldsymbol{x},\boldsymbol{v}}[y]$. With a slight abuse of notation, we can write it as $y^*(\boldsymbol{x})$ interchangeably (similar simplification will be used in the rest of the paper without causing confusion). In practice, however, we typically have no access to this ground truth response. Instead, we rely on a finite sample population $U = \{u_i\}_{i=1,\cdots,N}$ whose profiles $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N$ are drawn from the target distribution $\mathcal{T}$. Each individual u_i provides a response y_i. If we know $\overline{y}_i = \mathbb{E}_{y_i\sim\mathcal{Y}|\boldsymbol{x},\boldsymbol{v}_i}[y_i]$ which is considered as their rational decision since the operator $\mathbb{E}$ averages out the randomness of their decisions, we can adopt the average of these expected responses across the sampled population U, i.e., $y^{**} = \frac{1}{N}\sum_{i=1}^{N}\overline{y}_i$, as a gold-standard

approximation of y^*. However, the real human individuals' responses are typically noisy and can be expressed as $y_i = \overline{y}_i + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}(0, \eta_i^2)$ captures the individual randomness. Ideally, if we could collect individual responses repeatedly many times, we could accurately estimate the individual-level expected response $\overline{y}_i$, and consequently, obtain an accurate estimate of y^{**}. Nevertheless, in practice, an individual typically provides only a single noisy response to a specific problem, which hinders the estimation of $\overline{y}_i$. Therefore, we can only rely on the noisy response y_i, and substitute y^{**} with empirical mean $\overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$. $\overline{y}$ provides an unbiased estimate of y^{**}, which provides a ground truth (see the blue star in Figure 2.2). Recall that to build a digital population, CrowdLLM generates virtual individuals $\tilde{U} = \{\tilde{u}_i\}_{i=1,\cdots,N}$ that mirror the real human individuals U. Suppose the digital counterpart of individual u_i, denoted by $\tilde{u}_i$, is generated by CrowdLLM with the same profile $\boldsymbol{v}_i$. Their individual responses, either the expected $\overline{\tilde{y}}_i$ or the noisy $\tilde{y}_i$, can be linked to u_i, i.e., $\overline{y}_i$ or y_i, despite the deviations caused by any model's inherent limitations. Such a physical-digital pair is illustrated in Figure 2.2.

Following the decision-making process of CrowdLLM, we can simplify the notations and express the response to the problem $\boldsymbol{x}$ of the individual u_i generated by CrowdLLM as

$$\tilde{y}_i = y_{\text{ref}} + \delta(\boldsymbol{x}, \boldsymbol{v}_i) + \tilde{\varepsilon}_i,$$

where $y_{\text{ref}} = \mathbb{E}[\Phi(\boldsymbol{x})]$ is the reference decision generated by the LLM backbone Φ, $\delta(\cdot)$ denotes the belief generator, and $\tilde{\varepsilon}_i$ is the noise with $\mathbb{E}[\tilde{\varepsilon}_i] = 0$ and $Var[\tilde{\varepsilon}_i] = \tilde{\eta}_i^2$ which represents the inherent uncertainty of individual i. For simplicity, following Eq. (2.2), we only consider the blender is additive and $\mathcal{F}$ is normal. Then, we compare these responses with real humans' responses. We only consider the analysis of the average response for a specific problem $\boldsymbol{x}$ and compare $\overline{\tilde{y}}$ with the ground truth $\overline{y}$. Their discrepancy can be measured by a loss function $\ell(\cdot,\cdot)$ as $\ell(\overline{\tilde{y}}, \overline{y})$, e.g., here we focus on the squared loss to conduct our theoretical inquiry. The same proof strategies can be potentially extended to other loss functions, such as KL divergence. Inspired by the unified theory of diversity [215], we can prove the following theorem:

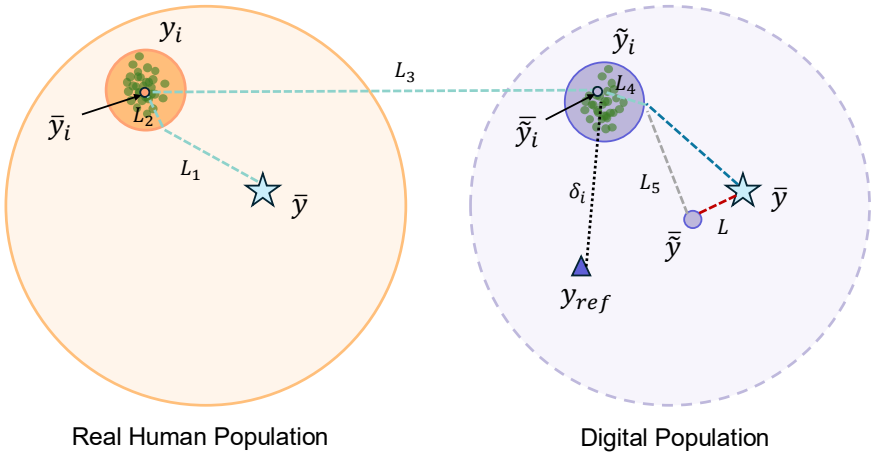


Fig. 2.2: An illustration of the risk decomposition for a specific problem $\boldsymbol{x}$. The yellow circle represents the sample human population U while the purple dashed circle represents their digital counterpart. The balls in the circles represent a physical human individual u_i and their digital counterpart $\tilde{u}_i$. $\overline{y}_i$ and $\overline{\tilde{y}}_i$ are the expected responses of the human individual and the digital individual, respectively. y_i and $\tilde{y}_i$ are their corresponding noisy observations. The empirical mean of the individual noisy responses $\tilde{y}_i$ across the whole digital population is represented by the light purple point $\overline{\tilde{y}}$. The dark purple triangle y_{ref} is the reference response generated by the LLM. The star $\overline{y}$ represents the average response of the sample population, adopted as a "ground truth". The five components L_1 to L_5 are explained in Theorem 2.

Table 2.1: A summary of loss terms in Theorem 2.3.2.

Loss	**Description**	**Practical Implications**
L_1	This term captures the loss induced by human individual-level biases relative to the population-level mean decision.	It arises from the heterogeneity of the real human population and can hardly be controlled. Its magnitude determines the difficulty of using an LLM to simulate human decision-making.
L_2	This term refers to the loss that arises from each human individual's behavioral stochasticity in decision-making.	It is intrinsic to human behavior and cannot be fully eliminated, but can be explicitly estimated via uncertainty quantification and mitigated by improved human–system interactions (e.g., appropriate task description).
L_3	This term characterizes the alignment between digital population and human population by quantifying discrepancies between each human individual and the corresponding digital counterpart.	It is a key measurement of the digital population's fidelity, and can only be reduced by accurate simulation of the real human individual beliefs.
L_4	This term captures within-individual variability in the digital population to account for the behavioral stochasticity of their human counterparts.	It can be controlled by adjusting variance-related parameters of the belief generator and the blender in CrowdLLM. However, such control should be exercised with care, as it directly affects individual variability, and if overly suppressed, leads to deterministic and overconfident decisions.
L_5	This term measures the diversity of the digital population, which depends on both the profile generator's ability to produce different individual profiles and the belief generator's capacity to generate diverse decisions.	All parameters of CrowdLLM that influence decision generation may affect this diversity. It must also be calibrated with care to ensure sufficient diversity without sacrificing others, as it might have interactions with other terms.

Theorem 2.3.2. *Consider a digital population $\tilde{U} = \{\tilde{u}_i\}_{i=1}^N$ generated by CrowdLLM with profiles $\mathcal{Z} = \{\boldsymbol{v}_i\}_{i=1}^N \sim \mathcal{T}$, and their real human counterparts $U = \{u_i\}_{i=1}^N$. Suppose the overall expected risk is $L = \mathbb{E}_{\mathcal{T}}\Big[\mathbb{E}_{\mathcal{D}}\big[\mathbb{E}_{\boldsymbol{x}\sim\mathcal{X},y\sim\mathcal{Y}}[\ell(\overline{y},\overline{\tilde{y}})]\big]\Big]$. Given the training data $\mathcal{D} = \bigcup_{i=1}^N \mathcal{D}_i$ where $\mathcal{D}_i$ is the data contributed by individual u_i, we have the following decomposition over the risk L:*

$$
\begin{aligned}
L = \mathbb{E}_{\mathcal{X}}\Big[& \underbrace{\mathbb{E}_{\mathcal{Z}\sim\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{y_i\sim\mathcal{Y}|\mathcal{X},\boldsymbol{v}_i}[\ell(\overline{y},y_i)]\big]}_{L_1:\ \textit{Average Human Bias}} \\
& + \underbrace{\mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\eta_i^2\big]}_{L_2:\ \textit{Human Individual Noise}} + \underbrace{\mathbb{E}_{\mathcal{T},\mathcal{D}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i,\overline{\tilde{y}}_i)\big]}_{L_3:\ \textit{Twin Discrepancy}} \\
& + \underbrace{\mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\tilde{\eta}_i^2\big]}_{L_4:\textit{Allowed Individual Uncertainty}} - \underbrace{E_{\mathcal{T},\mathcal{D}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{\tilde{y}},\tilde{y}_i)\big]}_{L_5:\textit{Digital Population Diversity}} \Big]
\end{aligned}
$$

Theorem 2 decomposes the expected risk of CrowdLLM into five parts that correspond to the average human bias, the human individual noise, the discrepancy between the individuals of the physical and digital populations, the allowed individual uncertainty, and the diversity of the digital population. When other components (L_1 to L_4) are fixed, Theorem 2 shows greater diversity of the digital population (L_5) will reduce the expected risk. Remarkably, the decomposition in Theorem 2 holds under the premise that the loss function $\ell(\cdot,\cdot)$ admits an ambiguity decomposition property (see Appendix A.2 Lemma 1). Thus, only certain loss functions (e.g., the squared loss discussed in Theorem 2) are sufficient to satisfy this property, while others may not. A well-representative class of such decomposable loss functions is given by the Bregman divergences [215].

Now we can similarly compute the decision-making risk for pure LLMs as:

Proposition 2.3.3. *With the same population $U = \{u_i\}_{i=1,\cdots,N}$ as in Theorem 2, pure LLM-based decision-making with zero-shot prompting yields the following decomposition over its*

risk $L' = \mathbb{E}_{\mathcal{T},\mathcal{X},\mathcal{Y}}[\ell(\overline{y}, y_{ref})]$*:*

$$L' = L_1 + L_2 + \mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i, y_{ref})\big] + \eta_\Phi(t),$$

where $\eta_\Phi(t) \geq 0$ *measures the randomness of the LLM outputs under temperature* t*.*

With Theorem 2.3.2 and Proposition 2.3.3, we can further provide a sufficient condition under which CrowdLLM outperforms pure LLM-based decision-making. Interestingly, this sufficient condition is built on the quality of the LLM backbones.

Assumption 2.3.4 (Quality of the LLM backbone)**.** Given any specific task $\boldsymbol{x}$, for $\alpha \in (0,1)$ and $\gamma \in (0,\alpha)$, there exists a constant κ_α, such that the deviation of the response given by the LLM backbone Φ from the gold-standard response $y^{**}(\boldsymbol{x})$ given by the human population is bounded with probability at least $1-\alpha$, i.e., $P(|\mathbb{E}[\Phi(\boldsymbol{x})] - y^{**}(\boldsymbol{x})| \leq \kappa_\alpha) \geq 1-\alpha$.

With a guarantee on the quality of the LLM backbone, we have the following theorem:

Theorem 2.3.5. *Consider a digital population generated by CrowdLLM* $\tilde{U} = \{\tilde{u}_i\}_{i=1,\cdots,N}$*. For a specific problem* $\boldsymbol{x}$*, assume the belief biases of* $\tilde{U}$ *is* $\delta_1, \cdots, \delta_N$*, with a mean* $\mu_\delta = \frac{1}{N}\sum_{i=1}^{N}\delta_i$ *and the second moment* $\varepsilon_\delta^2 = \frac{1}{N}\sum_{i=1}^{N}\delta_i^2$*. Suppose the deviation between the gold-standard response and the reference decision given by the LLM backbone is* $\Delta = y^{**}(\boldsymbol{x}) - y_{ref}$*. We can construct an interval*

$$\mathcal{B}_\alpha(\Delta) = \Big[\Delta - h_\Delta(\kappa_\alpha), \Delta + h_\Delta(\kappa_\alpha)\Big],$$

where

$$h_\Delta(\kappa_\alpha) = \begin{cases} h_1, & \textit{when } \dfrac{N-2}{\sqrt{2}N}A(N) \geq \kappa_\alpha \\ \\ h_2, & \textit{when } \dfrac{N-2}{\sqrt{2}N}A(N) < \kappa_\alpha, \end{cases}$$

$$h_1 = \frac{\sqrt{N^2\kappa_\alpha^2 + 2(N-1)[A(N)]^2} - (N-2)\kappa_\alpha}{2(N-1)},$$

$$h_2 = \frac{\sqrt{2}}{N} A(N),$$
$$A(N) = \sqrt{(N-2)\varepsilon_\delta^2 + N\eta_\Phi(t)},$$

such that when $\mu_\delta \in \mathcal{B}_\alpha(\Delta)$*, with probability at least* $1-\alpha$*, CrowdLLM leads to a smaller expected risk than its pure LLM-based counterpart, i.e.,* $L \leq L'$*.*

Theorem 2.3.5 indicates that to ensure that CrowdLLM outperforms the LLM backbone, the mean belief bias μ_δ should not be too far away from Δ (i.e., Δ quantifies the deviation of the reference decision by the LLM backbone to the ground truth). It is easy to see that the interval width merely relies on $h_\Delta(\kappa_\alpha)$. When the size of digital population N is small, $h_\Delta(\kappa_\alpha) = h_2$ does not depend on κ_α, but is instead directly affected by N. When N is large enough, $h_\Delta(\kappa_\alpha) = h_1$ is monotonically decreasing in κ_α, which means that if κ_α turns larger, we need a tighter interval that covers the belief bias to ensure $L \leq L'$. It suggests that if CrowdLLM is built with a lower-quality LLM backbone, the mean belief bias μ_δ needs to be closer to Δ, whereas a higher-quality LLM backbone will provide greater tolerance to guarantee CrowdLLM's performance. Moreover, the diversity of the digital population, reflected in ε_δ^2 and $\eta_\Phi(t)$, can somehow alleviate these constraints and offer even more capacity for CrowdLLM to surpass the LLM backbone. This theoretical analysis also reveals why LLM itself can't generate the needed diversity, since it is not just statistical derivations from a mean but needs to be productive in a specific context (e.g., which corresponds to a diverse distribution of participants' profiles). We know that an LLM can balance coherence and novelty in its responses by adjusting the temperature t, but from Theorem 3, we see that increasing t to improve novelty in responses can actually enlarge the gap between the LLM and CrowdLLM since it leads to more incoherence and noise of the backbone. In contrast, in CrowdLLM, more diversity associated with a larger ε_δ^2 might also result in an increase in $h_\Delta(\kappa_\alpha)$, which allows μ_δ to deviate more from Δ while ensuring the superiority of CrowdLLM. This answers Q3 as well.

We can further develop a confidence interval for CrowdLLM to cover the ground truth y^*. First, we restate the Theorem 1 in [216] in the context of our problem as follows:

Theorem 2.3.6. *Given any specific task $\boldsymbol{x}$, suppose $\theta^* \triangleq y^* = \mathbb{E}[y|\boldsymbol{x}]$ is the population mean response to be estimated. Consider a model f learned from the data to predict y. With $\alpha \in (0,1)$ and $\gamma \in (0,\alpha)$ fixed, suppose that for any possible θ, we can construct confidence sets $B^1_\gamma(\theta)$ and $B^2_{\alpha-\gamma}(\theta)$ satisfying $P(\mathbb{E}_{\boldsymbol{v}\sim\mathcal{T}}[f(\boldsymbol{x},\boldsymbol{v}) - y] \in B^1_\gamma(\theta)) \geq 1-\gamma$ and $P(\mathbb{E}_{\boldsymbol{v}\sim\mathcal{T}}[\theta - f(\boldsymbol{x},\boldsymbol{v})] \in B^2_{\alpha-\gamma}(\theta)) \geq 1-(\alpha-\gamma)$. Let $B^y_\alpha = \{\theta | \exists \theta_1 \in B^1_\gamma(\theta), \theta_2 \in B^2_{\alpha-\gamma}(\theta) \ \ s.t. \ \ \theta_1+\theta_2 = 0\}$. Then, we have $P(\theta^* \in B^y_\alpha) \geq 1-\alpha$.*

Inspired by this result, we can further show the following theorem:

Theorem 2.3.7. *Consider CrowdLLM with an LLM backbone Φ and a belief generator δ under an additive blender. Assume Φ has the randomness $\eta(t)$ that can only be changed through adjusting the temperature t. Fix $\alpha \in (0,1)$ and $\gamma \in (0,\alpha)$. Given any specific task $\boldsymbol{x}$, suppose we have n real human responses $y_1, \cdots, y_n$ taking numeric values for this task in the training data. Consider a digital population of size N generated by CrowdLLM, $\tilde{U} = \{u_1, \cdots, u_N\}$ with profiles $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N$. Here, we assume $\frac{n}{N} \to p \in (0,1)$. Suppose their decisions are $\tilde{y}_1, \cdots, \tilde{y}_N$, where $\tilde{y}_i = \Phi^{(i)}(\boldsymbol{x}) + \delta_i$ with $\Phi^{(i)}(\boldsymbol{x})$ being the ith response sampled from Φ and $\delta_i = \delta(\boldsymbol{x},\boldsymbol{v}_i)$ being the belief biases. Define $\sigma^2_\delta = \frac{1}{N}\sum_{i=1}^N (\delta_i - \overline{\delta})^2$ and $\sigma^2_r = \frac{1}{n}\sum_{i=1}^n (r_i - \overline{r})^2$ where $\overline{\delta} = \frac{1}{N}\sum_{i=1}^N \delta_i$, $r_i = y_i - \tilde{y}_i$ and $\overline{r} = \frac{1}{n}\sum_{i=1}^n r_i$. Suppose the training error can be bounded by a small tolerance ε_0^2, i.e., $\frac{1}{n}\sum_{i=1}^n (y_i - \tilde{y}_i)^2 \leq \varepsilon_0^2$. Then, we can build a confidence interval centered on the aggregated decision $\overline{\tilde{y}} = \frac{1}{N}\sum_{i=1}^N \tilde{y}_i$:*

$$B^y_\alpha = \left\{ y : |y - \overline{\tilde{y}}| \leq \varepsilon_0 + z_{1-\frac{\alpha}{2}} \sqrt{\frac{\eta(t)}{N} + \frac{\sigma^2_\delta}{N} + \frac{\sigma^2_r}{n}} \right\},$$

where $z_{1-\frac{\alpha}{2}}$ is the $1-\frac{\alpha}{2}$ quantile of the standard normal distribution, such that the ground truth y^* satisfies

$$\liminf_{n,N\to\infty} P(y^* \in B^y_\alpha) \geq 1-\alpha.$$

The above theorem indicates the effectiveness of CrowdLLM under asymptotic settings. From the interval built in this theorem, we notice the uncertainty in estimating y^* is mainly contributed by three parts: the randomness of LLM backbone, the variance of belief biases,

and the variance of the residuals σ_r^2. Since the LLM backbone is frozen and will only show a certain randomness controlled by the temperature t, the uncertainty from this component is fully contributed by $\eta(t)$. When N grows, the average converges and the uncertainty is gradually reduced. The uncertainty contributed by the belief generator is rooted in the belief biases. $\frac{\sigma_\delta^2}{N}$ measures the variability of the average belief bias. However, this uncertainty will not be explosively increasing, since the average belief bias will gradually converge to the true deviation of LLM's decision from the ground truth decision. As N grows, the diversity matters less and $\frac{\sigma_\delta^2}{N}$ is less influential. Beyond the first two sources of uncertainty, the quality of the belief generator determines how large the uncertainty will be. If the belief generator can well characterize the real human data, the individual residuals r_i should be small, which indicates the decisions given by CrowdLLM can well approximate the real human data, and σ_r^2 should be small when the individual residuals are consistently small. If n is large, the real human data used for training is large, which shrinks this variance and leads to a more accurate model. It should be mentioned that, although this theorem provides a way to construct confidence intervals that can somehow serve as a measure of the uncertainty in the collective decision-making of a digital population, it is not intended to provide a tight, finite-sample interval estimate that can be directly used as an exact plug-in uncertainty estimator by practitioners. In practice, we still recommend reporting the uncertainty depending on the specific task contexts with empirical measurements.

Remark. For a clear characterization of the theoretical properties of CrowdLLM, our analysis above primarily assumes a mean-score aggregation rule for decision-making under which the population-level decision is obtained by averaging individual decisions. In practice, this rule generally applies to continuous decisions and can also be used for discrete decision-making through appropriate transformation (e.g., by introducing continuous latent scores). However, it may not always be the preferred choice, and alternative aggregation rules are widely used for discrete decisions. A canonical example is majority voting (MV), a basic label aggregation approach that selects the label chosen by most participants [217, 218]. For binary decisions with $y_i \in \{0, 1\}$ for each individual $i = 1, \cdots, N$, the aggregated decision under

this approach can be written as $\overline{y} = \mathbb{I}[\frac{1}{N}\sum_{i=1}^{N} y_i \geq \frac{1}{2}]$, which is a non-smooth, threshold-based mapping of the empirical mean. As a result, this rule does not directly fall within the scope of the theoretical results in Theorems 2-5. Nevertheless, majority voting and other discrete aggregation rules can often be interpreted as operating on latent continuous scores (e.g., probabilities) followed by a discretization step. From this perspective, results on mean aggregation remain theoretically well-founded and somehow provide useful insights into these aggregation mechanisms. While rigorous guarantees under explicit non-linear or discrete aggregation rules are left for future work, we empirically evaluate the practical performance of CrowdLLM under several such rules in case studies to complement the theoretical analysis.

2.4 Case Studies

In this section, we conduct experiments to evaluate the ability of CrowdLLM to generate data of human-grade quality across different applications, including crowdsourcing [219], product reviews from users [220], and voting [221]. Each application entails a different kind of human data, but all involve a set of decision-making tasks that are attributed to a population of workers (in crowdsourcing), users (in recommendation systems), or voters (in politics). We evaluate CrowdLLM, its vanilla versions and some other benchmark methods, including LLM-based and non-LLM methods with a range of performance metrics such as accuracy, diversity, fidelity to real human data, sample efficiency, and cost. We also thoroughly study different configurations of CrowdLLM and its various components (i.e., how prompting strategies are employed to generate the LLM-based virtual human participants) and study its sample efficiency (i.e., how much training data is needed to reach a superior performance). Now, we introduce details of our experimental evaluations and main findings.

2.4.1 Experiment Setup

2.4.1.1 Overall Design and Evaluation Method.

Recall that CrowdLLM generates a collection of decisions from the synthetic participants for some given tasks, such as voting for alternatives, rating products, and evaluating texts. To

evaluate how well its outcome matches the data collected in a real human population, we assess its performance in both aggregated decision (accuracy) and the distribution of decisions (diversity). Specifically, we use *Average Wasserstein Distance (Avg. WD)* to measure the average of the distributional deviation from the gold standard across each test problem. A lower value of this metric indicates better distributional similarity and a more effective characterization of population diversity. For the evaluation of aggregated decisions, we consider evaluation metrics such as *Mean Absolute Error (MAE)*, *Root Mean Square Error (RMSE)*, and *Cosine Similarity (CS)*. The first two metrics emphasize the average performance over different tasks, while CS focuses on the general comparison with the gold standard (i.e., real human data) on the test set. For Crowdsourcing tasks, we follow prior work [40] and adopt the common practice of considering various aggregation approaches that include mean for all case studies, and also more specialized ones for crowdsourcing applications such as MV, Dawid-Skene (DS), and Generative model of Labels, Abilities, and Difficulties (GLAD) [222]. Different from simple MV, DS improves aggregation by learning how accurate each participant is from their labeling history through a probabilistic formulation and giving greater weights to reliable ones. GLAD goes further by also considering task difficulty, using a probability model to combine participant's ability and task difficulty for robust label estimation.

2.4.1.2 Implementation of CrowdLLM.

Unless otherwise specified, we use Gemma3-12B [191] as the LLM backbone of CrowdLLM due to its strong performance, its reliable capabilities across diverse tasks, and manageable size which facilitates extensive experimentation. In each of our case studies, we also conduct a comparison of Gemma3-12B with three other prominent LLMs, Deepseek-Distill-R1-Llama-8B (Deepseek R1) [192], Llama3-8B-lexi-uncensored [223], and Qwen3-8B [224], and found Gemma3-12B indeed outperforms others. In all the experiments, we set the temperature to 0 for CrowdLLM, and set the reference generation parameter K to 8, the offset parameter J for personalized decision-making in training to 10, and the regularization controller λ to 1. These settings are based on our extensive empirical experiments across datasets which consistently

provide good performances of CrowdLLM. For all the experiments, the candidate profile pool $\mathcal{S}$ is built on the historical user data and the profile generators are implemented as probabilistic samplers on this pool built from the user data as well. For the belief generator, we use a VAE-style model as mentioned before to implement this component and set the number of latent dimension for its encoder to 20, following the prior work [56, 225] to ensure the balance between representational capacity and computational cost. The text encoder of all problem descriptions g_x is implemented by a pre-trained sentence transformer, text-embedding-nomic-embed-text-v1.5 [226]. The profile encoder g_z maps the individual profiles into embeddings that can be directly fed into the generator. For cases where user profiles are built on their characteristics (e.g., *Crowdsourcing* and *Voting*), g_z is implemented by mapping each attribute of the profile vector into a one-hot vector and concatenating them as a profile-specific embedding. In the case of *Product Rating*, the profiles are readily usable embeddings, and thus g_z reduces to an identity mapping as no additional transformation is required. We use Adam [227] as the optimizer with a learning rate 0.001 to train CrowdLLM with Eq. (2.4). For all three cases, the implicit prior distribution q_ϕ is parametrized by a two-layer MLP that takes a Gaussian noise in the input, while the explicit $q_{\boldsymbol{\beta}}$ adopts a Gaussian distribution. Across all the experiments, the data are preprocessed to ensure compatibility with the choice of ℓ as a squared loss. Another important aspect of implementing CrowdLLM on a particular application is the profile generation. Recall that the profile generator aims to generate diverse profiles for the synthetic human participants. The individual profile are used as the contextual information which is further fed into the belief generator of CrowdLLM. For a given application, one can either obtain summary statistics of the human participants or design an ideal distribution of the profile variables that are representative of a real crowd. Figure 2.3 shows an example of such a distribution of participant profiles based on 5 variables: *Gender*, *Age*, *Race*, *Occupation*, *Education* that we used in Case Study I.

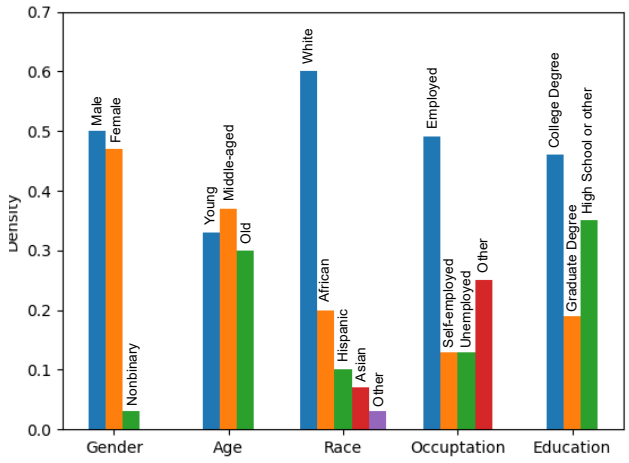


Fig. 2.3: An example of the distribution of participants' profiles.

2.4.1.3 Baselines.

We adopt a variety of prompting strategies to create representative baselines of LLM-based synthetic crowd. This includes the zero-shot (direct) prompting, referred to as LLM (zero-shot), which prompts the LLM to generate decisions on problems with the most basic information. We also include multi-persona prompting [228, 229] and self-consistency (SC) [230]. Multi-persona prompting involves a collaboration of multiple LLMs prompted with different profiles to create diverse personas. Self-consistency prompting is performed by taking the majority vote of multiple decisions generated by LLM (i.e., with temperature set at 0.5) following the practice in [230, 231]. More specifically, the zero-shot prompt provides LLM with problem description $\boldsymbol{x}_t$, specific requirements $\mathcal{R}_t$ on the decisions, and context $\mathcal{C}_t$. Here, $\boldsymbol{x}_t$ describes the problem that the participants need to make decisions on. $\mathcal{R}_t$ describes what kind of decisions need to be made, e.g., if the decision is a score, the scale of the score should be included in $\mathcal{R}_t$. $\mathcal{C}_t$ encodes information about the decision-making scenario. An example of the zero-shot prompt is shown in the Appendix. For multi-persona prompting, i.e., using multiple personas in prompting for self-collaboration as shown in [232, 229], we build multiple personas with additional context on their profiles. We use similar prompts as zero-shot

prompting for making decisions, but change the context $\mathcal{C}_t$ to assign persona. An example of a multi-persona prompt is shown in the Appendix. For self-consistency prompting, we follow the strategy adopted in [230, 231] by taking the majority vote of multiple decisions generated by LLM with temperature 0.5. For the generation of each decision, we use the same prompt as zero-shot prompting. We can certainly adopt more prompting strategies, such as the CoT prompting in [233]. In our experiments, we found no significant differences among the prompting strategies, and our focus is not on what the best prompting strategies in LLMs are to simulate humans, but the integration of those LLM-based frameworks with generative models, so throughout our experiments, we use zero-shot, multi-persona, and SC prompts.

2.4.2 Case Study I: Crowdsourcing

We evaluated CrowdLLM and the other baselines using two publicly available crowdsourcing datasets, *Offensiveness Rating* and *Question Answering Difficulty* [234]. The *Offensiveness* dataset contains 13,036 instances annotated by 263 workers across 1,500 problems to identify offensive text. *QA Difficulty* includes 4,576 instances annotated by 458 workers of 1,000 problems to assess question-answer pair difficulty. For each dataset, we use responses from 80% of the distinct workers as the training set and reserve the remaining 20% of workers' responses for testing. Tables 2.2-2.3 show that CrowdLLM consistently outperforms other baselines by achieving the lowest Avg. WD and the highest CS. This shows that our CrowdLLM can better capture real human diversity. We can also see that by incorporating diverse profile information to allow for more variability in decisions, even LLM (multi-persona) shows better performances compared to LLM (zero-shot), though the improvement is not as substantial as in CrowdLLM. And pure generative models without LLM (i.e., the VAE model) often outperform the baseline LLM (zero-shot) method in terms of Avg. WD, highlighting the power of generative models in diversifying predictions that improve CrowdLLM's alignment with real humans' responses. In terms of the aggregated decision, CrowdLLM generally offers competitive or improved MAE/RMSE compared with other approaches. On both *Offensive-*

ness and *QA Difficulty* datasets, CrowdLLM's MAEs are notably better than the baseline LLM methods under most of the aggregation methods.

Table 2.2: Performance Comparison on Crowdsourcing: Offensiveness Rating

Method	MAE↓					RMSE↓					CS↑	Avg. WD↓
	Mean	Median	MV	DS	GLAD	Mean	Median	MV	DS	GLAD		
Random	1.27	1.61	1.57	1.73	1.89	1.44	1.86	2.05	2.17	2.31	0.56	1.31
LLM (zero-shot)	0.86	1.03	1.23	1.06	1.23	1.08	1.32	1.53	1.38	1.52	0.39	1.13
LLM (multi-persona)	0.65	0.64	0.67	**0.69**	0.68	0.86	1.00	1.16	**1.15**	**1.16**	0.69	0.78
LLM (SC)	0.99	1.26	1.48	1.22	1.49	1.16	1.48	1.68	1.52	1.69	0.34	1.24
VAE	0.71	0.81	0.60	0.96	0.92	0.94	1.15	1.24	1.37	1.54	0.76	0.79
CrowdLLM	**0.45**	**0.48**	**0.43**	1.00	**0.53**	**0.59**	**0.82**	**1.10**	1.53	1.23	**0.85**	**0.51**

LLM backbone: Gemma 3-12B.

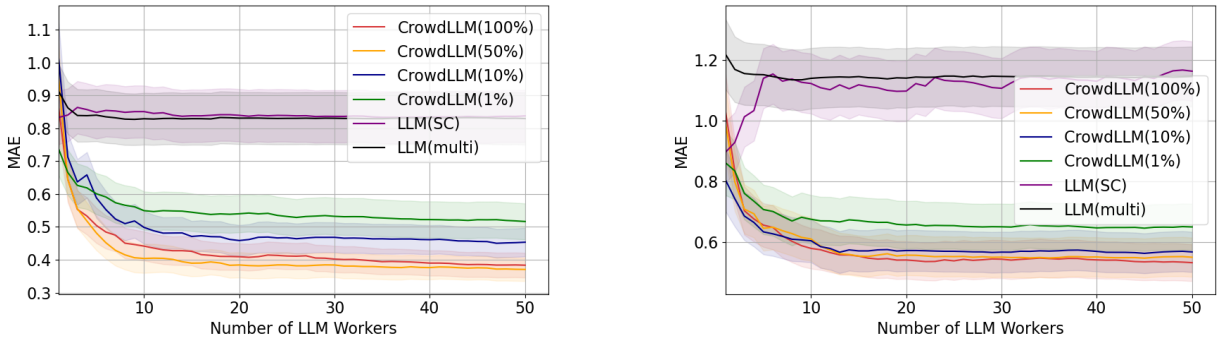
Table 2.3: Performance Comparison on Crowdsourcing: QA Difficulty

Method	MAE↓					RMSE↓					CS↑	Avg. WD↓
	Mean	Median	MV	DS	GLAD	Mean	Median	MV	DS	GLAD		
Random	1.21	1.41	1.52	1.98	1.81	1.43	1.73	2.02	2.38	2.26	0.49	1.29
LLM (zero-shot)	0.71	0.81	1.04	1.02	1.05	0.90	1.04	1.24	1.28	1.26	0.33	1.06
LLM (multi-persona)	0.73	0.82	1.00	1.06	1.02	1.02	1.16	1.38	1.46	1.41	0.48	0.93
LLM (SC)	0.68	0.79	1.02	**0.98**	1.03	0.87	**1.02**	1.22	**1.23**	**1.25**	0.36	1.01
VAE	0.76	0.90	0.71	1.41	1.11	0.98	1.20	**1.14**	1.85	1.53	0.68	0.88
CrowdLLM	**0.65**	**0.74**	**0.63**	1.49	**0.89**	**0.83**	1.06	1.18	2.08	1.43	**0.71**	**0.74**

LLM backbone: Gemma 3-12B.

One might be interested in how much real human data is needed to train CrowdLLM well. To answer this question, we further conduct some computational experiments and show the results in Figure 2.4. Specifically, for each task/instance in the crowdsourcing case studies, we incrementally add more workers and monitor CrowdLLM's performance. Figure 2.4 shows that the performance of CrowdLLM (i.e., evaluated by MAE) consistently improves as the number of workers increases. Note that in Figure 2.4 the labels, CrowdLLM (x%), mean the

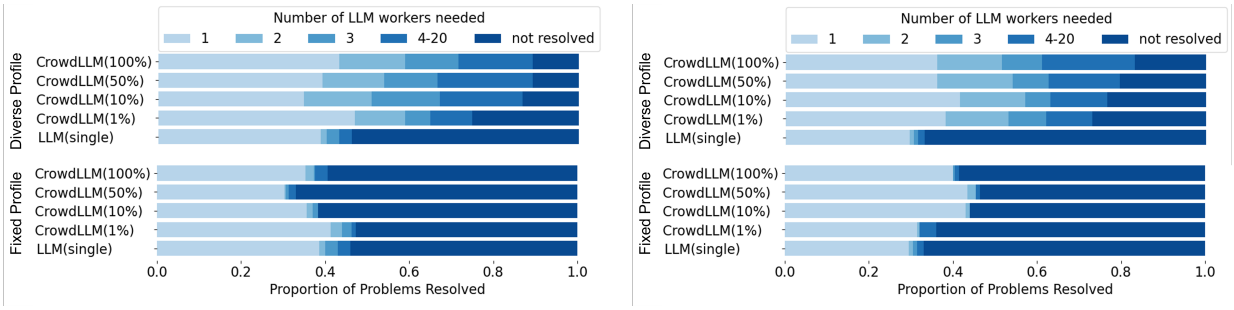
model is trained with x% of the human workers' data. In contrast to the low diversity and high uncertainty exhibited by LLM baselines, it is impressive to see that, even with only 1% of the human workers which collectively provided around 100 responses in the training data, CrowdLLM can achieve the level of performance comparable to the full-data setting where



(a) Offensiveness dataset

(b) QA Difficulty dataset

Fig. 2.4: MAE with increasing simulated workers across training worker sizes; CrowdLLM (x%) means the model is trained with x% of the real human workers' data



(a) Offensiveness dataset

(b) QA Difficulty dataset

Fig. 2.5: Resolution rate with increasing simulated workers in CrowdLLM under diverse and fixed profiles; CrowdLLM (x%) means the model is trained with x% of the human workers' data

CrowdLLM is trained with all the training data (100%), highlighting the data efficiency and cost-saving potential of CrowdLLM.

Another question one may ask is how many virtual human workers CrowdLLM needs to generate to resolve the problems (i.e., reducing absolute error below 0.5)? We conduct more experiments as well to answer this question and show the results in Figure 2.5. We can see that, across the different levels of the training size, the resolution rate increases steadily as the number of virtual workers with diverse profiles grows, whereas on the other hand, if we fix the worker profiles, it leads to a significant degradation. It underscores the value of promoting diversity among the virtual workers, which is a core strength of CrowdLLM. Recall that in the overall design of CrowdLLM, profiles serve as proxies for workers' beliefs: different profiles correspond to different backgrounds and thus to diverse beliefs about a task. When the virtual workers have diverse profiles (i.e., by generating ε_i following Eq. (2.3)), their varying beliefs allow the collective answer to be iteratively refined, and one can see in Figure 2.5 that the resolution rate increases steadily as the number of virtual workers grows. In contrast, under fixed profiles (i.e., by setting ε_i in Eq. (2.3) as a fixed number across individuals), the virtual workers hold nearly identical beliefs, so adding more of them does not improve the solution. This result highlights that the performance improvement of CrowdLLM comes more from the diverse profiles rather than simply from having more virtual workers.

2.4.3 Case Study II: Product Ratings by Users

In this subsection, we showcase the effectiveness of CrowdLLM on generating product reviews that can be used to train recommendation systems. We consider *Amazon Beauty* and *Amazon Music*, two datasets extracted from the Amazon Reviews 2023 dataset [235, 236]. *Amazon Beauty* is the subset of "All Beauty" category in which each product has been reviewed by 20 to 30 distinct users. It contains 448 products, with a total of 11,154 reviews from 10,957 unique users. *Amazon Music* is a filtered subset of the "Musical Instruments" category containing products reviewed by exactly 20 distinct users. It comprises 629 prod-

ucts, encompassing 12,580 reviews written by 12,396 unique users. For both datasets, we perform necessary preprocessing steps and hold out 20% of the full data as a test set based on unique problem IDs. All the experimental results are reported based on the held-out test set. The objective of CrowdLLM is to generate the distribution of ratings of each product by providing its product information to the digital population CrowdLLM creates. Results of CrowdLLM and the other methods are shown in Tables 2.4 and 2.5. All methods use the same product information. LLM multi-persona additionally conditions on user-evaluation text to simulate more diverse responses of the virtual users. CrowdLLM also incorporates the user-evaluation text into its training process. We can see in Tables 2.4 and 2.5 that on both datasets, CrowdLLM achieves the best performance in terms of all the evaluation metrics. We further show in Figures 2.6 and 2.7 the generated distributions of the product ratings by CrowdLLM and the other methods (i.e., we randomly selected three products from each of the two datasets), together with the distribution of real human users. We can see that LLMs, even when we adjust temperature or provide user-evaluation text in multi-persona prompting, still exhibit limited diversity in their generated ratings. VAE, by contrast, produces more diverse outputs but falls short of accurately matching true human ratings. CrowdLLM achieves a better balance: it captures both diversity and accuracy, resulting in rating distributions that closely align with those of human participants.

Table 2.4: Performance Comparison on Recommendation(All Beauty)

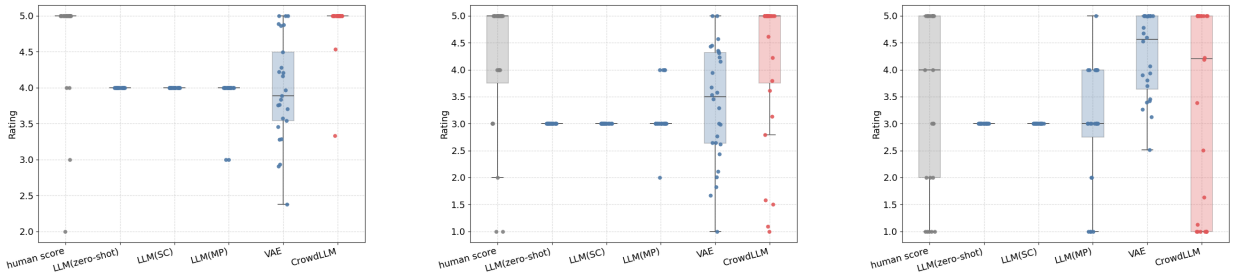
Method	MAE↓	RMSE↓	CS↑	Avg. WD↓
random	1.19	1.30	0.59	0.14
LLM (zero-shot)	1.04	1.19	0.13	0.15
LLM (multi-persona)	0.87	0.97	0.24	0.09
LLM (SC)	1.03	1.16	0.13	0.14
VAE	0.88	1.02	0.28	0.06
CrowdLLM	**0.21**	**0.26**	**0.93**	**0.04**

LLM backbone: Gemma 3-12B.

Table 2.5: Performance Comparison on Recommendation(Musical Instruments)

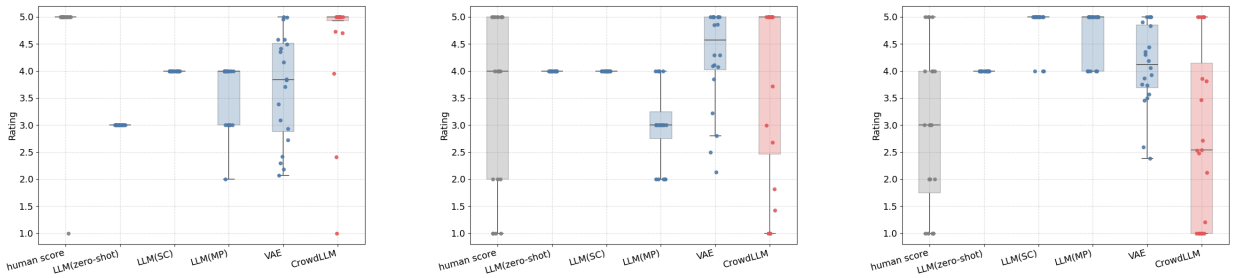
Method	MAE↓	RMSE↓	CS↑	Avg. WD↓
random	1.28	1.39	0.58	0.15
LLM (zero-shot)	0.52	0.65	0.27	0.16
LLM (multi-persona)	0.47	0.58	0.37	0.14
LLM (SC)	0.46	0.57	0.29	0.16
VAE	0.47	0.61	0.63	0.07
CrowdLLM	**0.22**	**0.27**	**0.92**	**0.05**

LLM backbone: Gemma 3-12B.



(a) (b) (c)

Fig. 2.6: Rating scores of three randomly selected products from Amazon Beauty



(a) (b) (c)

Fig. 2.7: Rating scores of three randomly selected products from Amazon Music

2.4.4 Case Study III: Voting

Table 2.6: Performance Comparison on Voting

Method	MAE↓	RMSE↓	CS↑	Avg. WD↓
random	5.00	6.55	0.72	4.00
LLM (zero-shot)	11.92	15.41	0.38	9.00
LLM (multi-persona)	10.00	13.69	0.40	6.50
LLM (SC)	11.42	14.97	0.35	7.92
VAE	11.67	14.88	0.32	7.67
CrowdLLM	**4.00**	**5.02**	**0.83**	**1.67**

LLM backbone: Gemma 3-12B.

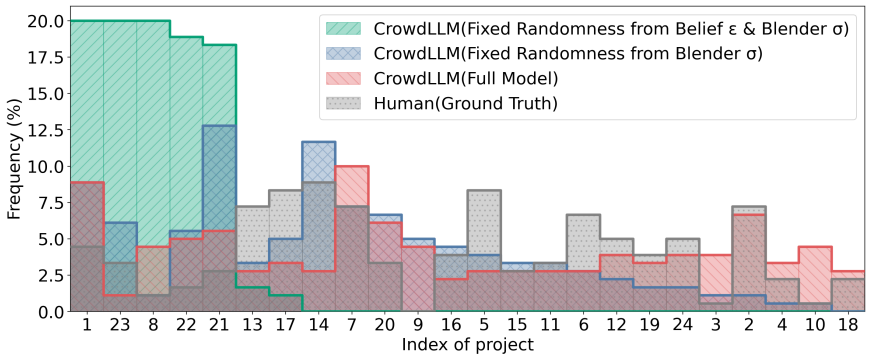


Fig. 2.8: Voting migration across models: from fixed belief with blender to relaxed belief and full CrowdLLM — increasing diversity with subsequent accuracy refinement

We further evaluate CrowdLLM and other methods on a voting dataset. This Zurich PB Voting dataset [54] was conducted in March 2023 with 180 participants where each participant evaluated 24 projects and expressed preferences through several predefined preference selection methods. Existing work has developed LLM-based synthetic crowds to replicate

the voting outcomes in this dataset, such as [54]. One problem found in these studies is that, despite the great promise of LLMs in generating human voting results, there is a lack of diversity as LLM-generated votes tend to concentrate heavily on only a few voting options, leaving many other alternatives with no votes at all, as reported in [54]. This is a significant shortcoming of the LLM-based virtual voters. Therefore, in this case study, we aim to evaluate CrowdLLM using the same data studied in [54]. In our experiments, we adopted the top-five selection method, where each participant selects five preferred projects. The dataset also contains participant-specific preference information, such as project location, topic, and cost, which is incorporated into our modeling. In summary, the dataset contains 180 voting records (five selections each), and we split it with 80% for training and 20% for testing. We use LLMs, VAE, random baseline, and CrowdLLM to create a simulated voter population. In each experiment, each virtual vote generates a set of five preferred projects. We then aggregate these generated votes into per-project vote counts and compare them with the real human vote counts, evaluating performance using MAE, RMSE, CS, and WD. Results are shown in Table 2.6 which shows that CrowdLLM achieves the best results across these metrics. In terms of diversity, LLMs and VAE tend to concentrate on a few projects, with limited variation across participants' preferences. We also conduct an ablation study to show how different components of CrowdLLM impact its result. Figure 2.8 illustrates the voting results of different configurations of the CrowdLLM. Under fixed beliefs, i.e., CrowdLLM (fixed randomness from belief ε and blender σ), voters cast in their votes in the same way, which is similar to the behavior of LLMs. By allowing each voter's belief to be randomly generated, we can see that CrowdLLM (fixed randomness from blender σ) improves diversity, yet still there are several projects with very few votes. In contrast, the full CrowdLLM model not only preserves diversity but also accurately captures the few votes for the less popular projects, leading to greater consistency with real human voting patterns.

2.4.5 Performance Study Using Simulated Datasets

The three real-world case studies demonstrated CrowdLLM's effectiveness in generating digital populations that have high fidelity to real human populations. In this subsection, we aim to further study which factors of the training dataset mostly impact CrowdLLM's effectiveness. These factors concern the signal-to-noise level of the data, the sample size, etc. We generate datasets by different combinations of four factors: the number of participants, the number of tasks assigned to each participant, the accuracy of participants' responses, and the diversity of participants' beliefs. We evaluate CrowdLLM's performance by varying these factors together with changing the signal-to-noise level and sample size of the training data, etc.

2.4.5.1 Simulation Design.

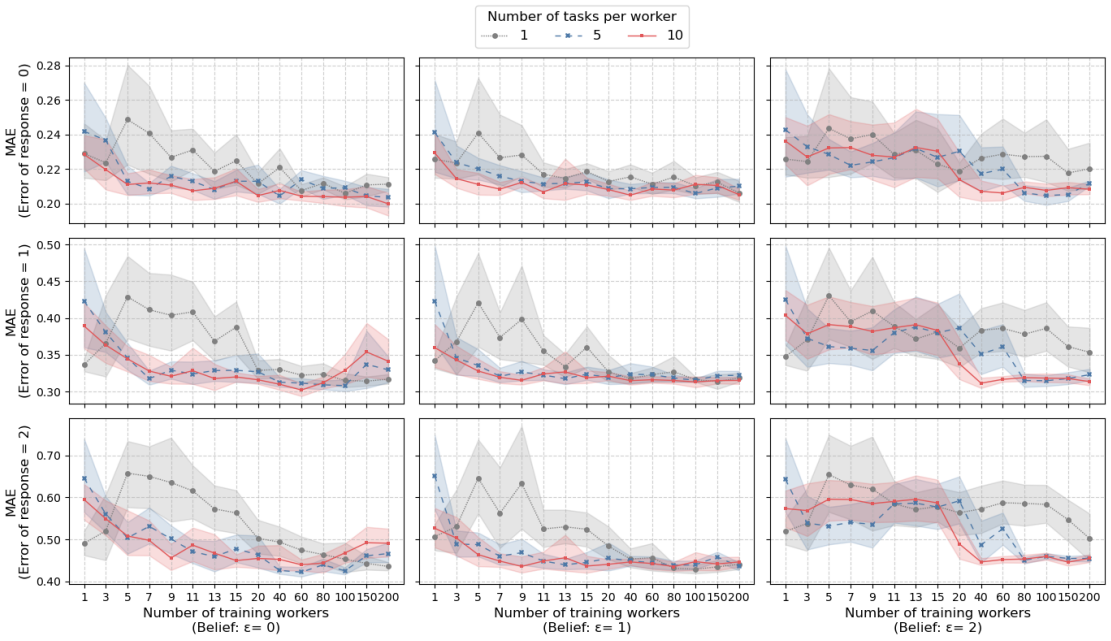


Fig. 2.9: Simulation studies across different signal-to-noise levels of the simulated datasets

Without loss of generality, we use the CrowdLLM model trained on the Offensiveness dataset in Case Study I as a ground truth model. In other words, we use the digital population created by CrowdLLM, which is trained on the Offensiveness dataset as the "real" population in this simulation study. In this way, we have the ground truth of the simulated datasets and can accurately evaluate the performance of CrowdLLM and other models. Then we generate datasets by different combinations of four factors: the number of participating workers, the number of tasks (i.e., the problems to solve) assigned per worker, the accuracy of worker responses, and the diversity of workers' beliefs. Each dataset is generated according to a different design, and the dataset is randomly divided into training and testing partitions, with 20% of the tasks held out for testing. CrowdLLM is trained on the training set, and is then evaluated on the held-out test set. To account for randomness, each experiment is repeated 10 times with independent initializations. During testing, 20 virtual workers are instantiated in CrowdLLM for each question, and their responses are averaged to form the final answer. To manipulate the signal-to-noise levels of the simulated datasets by the CrowdLLM model, we further add Gaussian noise to the generated data. We consider three degrees of noise (i.e., standard deviation = 0, 1, 2, as indicated by the error of response). These three settings correspond to the first, middle, and last rows of Figure 2.9, respectively. Within each accuracy scenario, we evaluate how the four factors, i.e., the belief diversity, the number of participating workers, and the number of questions per worker, impact the performance of CrowdLLM. The factor, belief diversity, is modeled using Gaussian distributions with standard deviations of 0, 1, and 2. We can see in Figure 2.9 how the increasing of the number of workers versus the increasing of the number of tasks per worker affects the performance of CrowdLLM (i.e., evaluated by MAE), and how different levels of belief diversity shift the trade-off between accuracy and cost, as the three columns in Figure 2.9 represent different degrees of belief randomness ($\varepsilon = 0, 1, 2$). For each experimental setting, the horizontal axis represents the number of workers used in the training data of CrowdLLM, while the color shade of the lines indicates the number of tasks per worker.

2.4.5.2 Key Insights from the Simulation Studies.

In summary, we found five key insights from Figure 2.9 that clarify the roles of belief diversity, worker numbers, and workload in determining CrowdLLM's performance under varying signal-to-noise conditions of the simulated datasets. First, when the simulated datasets contain considerable noise (error of response =1 or 2), if we set belief diversity to 0, adding more workers in the training data does not improve CrowdLLM's performance. Instead, the uniformity of beliefs causes errors to reinforce one another, and the MAE increases as the number of workers increases. This is shown in the second and third rows of the first column in Figure 2.9. Second, if datasets are noise-free (error of response =0), we should set belief diversity to 0 such that CrowdLLM achieves the best performance. However, in practice, this is an impossible scenario where everyone is perfectly aligned on the same response. This is shown in the first figure in Figure 2.9. Third, when belief diversity is high (i.e, set to 2), CrowdLLM demands a large annotation budget to reach good performance. This is shown in the last column of Figure 2.9, i.e., we need roughly 800 annotations (i.e., the product of the 20 virtual workers, each responding to 40 problems during testing) before MAE begins to stabilize. This pattern holds true across all levels of response quality, showing that high diversity systematically raises the annotation cost required to achieve reliable performance, as fewer workers require each to respond to more tasks to reach stability. Fourth, in the moderate diversity setting (belief diversity = 1), CrowdLLM achieves a more efficient balance between the number of workers and the number of tasks responded per worker. As shown in the middle column of Figure 2.9, roughly 10 workers (each responds to 5 or 10 tasks) can achieve strong performance, which is more efficient than the high-diversity setting that requires about 800 total annotations. Fifth, across all scenarios, the best performance that CrowdLLM can achieve is ultimately limited by the overall quality of the dataset. Datasets with more accurate worker responses provide better guidance for the model, leading to better performance of CrowdLLM. In summary, the quality of the dataset sets performance ceiling for CrowdLLM. To approach this ceiling efficiently, other factors must be carefully balanced.

In particular, moderate belief diversity achieves a favorable trade-off between the number of workers and the number of tasks each worker responds to in the training data, yielding strong performance with relatively few annotations.

2.5 Conclusion

This paper introduces CrowdLLM, a novel framework that leverages lightweight generative models with LLM-based virtual crowdworkers to emulate the decision-making diversity and distributional fidelity typically observed in a range of crowd-based decision-making applications such as crowdsourcing, voting, and product review. Empirical evaluations across real-world and simulated datasets demonstrate that CrowdLLM achieves promising performance in both accuracy and distributional fidelity to human judgments. CrowdLLM outperforms strong baselines and remains robust under data scarcity. In future work, we plan to further refine CrowdLLM with more specialized mechanisms for particular application contexts, e.g., by integrating with human behavior models and choice models, or characterize the data-generating process in finer details.

2.6 Further Discussion

2.6.1 CrowdLLM and Emerging Paradigms of LLM Use

It should be emphasized that CrowdLLM should not be viewed as a direct replacement for supervised fine-tuning, RLHF, or other related alignment methods. Rather, it represents a different way of using LLMs. Fine-tuning methods, as well as more recent on-policy distillation approaches [237, 238], are model-centric approaches that directly adapt a single LLM to specific tasks using either demonstrations or self-generated trajectories, while largely retaining the broad knowledge established through pretraining. RLHF, on the other hand, uses human feedback or learned reward signals to align the LLM's outputs with human preferences. Although it differs from fine-tuning in its training objective and learning process, RLHF is still model-centric in the sense that the goal is to shape the behavior of a single LLM. In contrast, CrowdLLM is more population-centric. It does not seek to turn a single

LLM into an optimal standalone decision-maker, but keeps it as a shared reference generator combined with generative components to build a broader population-generation framework. The goal is therefore not only to improve answer quality, but to emulate the heterogeneous and collective decision-making behaviors of a real human population. This distinction is especially important for crowd-based decision-making, where quality is not measured by a single aligned model response, but across multiple dimensions at both individual and population levels.

Recent progress in LLM-based systems has increasingly shifted beyond these model-centric approaches toward harnessing them as flexible components in larger computational workflows. In many applications, LLMs are connected with retrieval modules, memory, external tools, or multi-agent orchestration. This trend reflects an important realization: the value of LLMs often comes not only from the raw model itself, but also from how the model is embedded, constrained, and integrated within a broader system. CrowdLLM is consistent with this emerging view in the sense that it does not treat a single LLM as a self-contained solution. However, in addition to their different objectives, the core mechanism of CrowdLLM is also fundamentally different from harnessing-based systems. CrowdLLM still requires human data for human-centered calibration, rather than relying merely on prompt-based agentic pipelines that primarily improve operational capability. A promising future direction is to enable a combination of these two paradigms by integrating the generative calibration component of CrowdLLM with selected harnessing techniques. This would extend the component beyond a purely data-driven process toward more agentic, personalized adaptation. In this way, CrowdLLM could evolve from a reference-and-calibration scheme into a more integrated framework for building digital populations. While preserving the overall population-generation structure, each digital individual could be equipped with its own context, memory, data, retrieval sources, tools, and decision rules, which enables richer reasoning and interaction capabilities at the individual level. Therefore, the resulting digital population would better reflect the characteristics of real human populations at both individual and population levels, allowing individual behaviors and collective decisions to be

jointly analyzed and updated over time.

2.6.2 Exploratory Generation and the Evaluation Gap

The power of digital populations lies in their ability to provide useful simulations for early-stage analysis and decision-making by characterizing the behaviors of real humans, even for users not directly observed. However, generating a realistic digital population relies on the capacity of CrowdLLM to generalize beyond limited calibration data and capture plausible forms of human heterogeneity that have not been fully explored. This raises a natural dilemma: the goal of creating a digital population is to approximate the target real human population without fully observing it. Yet if the target population is not already known, how can we determine whether the digital population created by CrowdLLM is meaningful?

In our empirical analysis, we assume access to the profiles of a target population in the test set and focus only on evaluating the fidelity of the decision-making behaviors generated for those profiles. But in real-world settings, profiles of the target population are typically not fully known in advance. Thus, the problem is not only to generate realistic decisions conditional on given profiles, but also to determine which profiles should be included and how the resulting digital population should be evaluated. Although we may use the profile generator to randomly sample profiles that meet task-specific requirements, this alone does not guarantee representativeness when the candidate pool is biased, incomplete, or misaligned with the unknown target population. In such cases, high decision-making fidelity on the digital population may not imply alignment with the true population of interest. On the other hand, even when CrowdLLM identifies underexplored user groups with reasonable profiles, limited evidence may still make its simulated results difficult to trust.

Together, these challenges reveal an important evaluation gap in exploratory digital population generation. To close this gap, future work should develop systematic evaluation frameworks that provide evidence for both the adequacy of digital population profiles (e.g., coverage, balance, sensitivity, etc.) and the fidelity of decision-making behaviors of the digital population. Possible directions include stress-test benchmarks, population alignment tests,

uncertainty quantification measures, and selective human-validation protocols that identify which user segments, tasks, or response patterns should be checked with real human data.

2.7 Appendix

2.7.1 Proofs of Theoretical Results

2.7.1.1 Proof of Theorem 1

Proof. First of all, following the Theorem 1 in [213], we see that for any $\epsilon \in (0,1)$ and continuous distribution $\tilde{\mathcal{T}}$, there exists a generative model G such that

$$W_1(G_\sharp \rho, \tilde{\mathcal{T}}) < \epsilon,$$

where W_1 is the 1-Wasserstein distance. It indicates that we can always find a generative model that approximates any target continuous distribution. Next, we extend the result to arbitrary distribution with mixed-type data. Denote two sets S_1 and S_2. For a vector $\boldsymbol{x} = (x_1, \cdots, x_d)$, let's denote that for $\forall i \in S_1$, x_i is continuous, while for $\forall i \in S_2$, x_i is discrete. Here $S_1 \bigcup S_2 = \{1, \cdots, d\}$ and $S_1 \bigcap S_2 = \varnothing$. Now we show how to replace all the discrete variables by continuous variables and generate new distributions. Suppose for any $j \in S_2$, x_j follows a K-valued discrete distribution $f(x) = \sum_{k=1}^{K} p_k \delta(x - v_k) \triangleq P_j$ where δ is a Dirac measure, v_k is the k-th value of x_j and $\sum_{k=1}^{K} p_k = 1$. Consider $\tilde{\boldsymbol{x}} = (x_1, \cdots, \tilde{x}_j, \cdots, x_d)$ where $\tilde{x}_j \sim \sum_{k=1}^{K} p_k \mathcal{N}(v_k, \varepsilon^2 \eta^2) \triangleq Q_j$ where η is a fixed constant. Thus, based on the additive property of Wasserstein distance, we can obtain

$$W_1(P_j, Q_j) \leq \sum_{k=1}^{K} p_k W_1(\delta(v_k), Q_j^{(k)}), \tag{2.5}$$

where $Q_j^{(k)} \triangleq \mathcal{N}(v_k, \varepsilon^2 \eta^2)$. It is easy to see that

$$W_1(\delta(v_k), Q_k^{(j)}) = \mathbb{E}_{x \sim Q_k^{(j)}} |x - v_k| = \mathbb{E}_{x \sim \mathcal{N}(0, \varepsilon^2 \eta^2)} |x| = \sqrt{\frac{2\eta^2}{\pi}} \varepsilon.$$

Thus, with Eq.(2.5), we have

$$W_1(P_j, Q) \leq \sum_{k=1}^{K} p_k \sqrt{\frac{2\eta^2}{\pi}} \varepsilon = \sqrt{\frac{2\eta^2}{\pi}} \varepsilon.$$

where $j \in S_2$. It further gives $W_1(F(\boldsymbol{x}), F(\tilde{\boldsymbol{x}})) \leq \sqrt{\frac{2\eta^2}{\pi}}\varepsilon$. By running in all $j \in S_2$, we can build a sequence $\boldsymbol{z}_0, \boldsymbol{z}_1, \cdots, \boldsymbol{z}_{|S_2|}$, where $\boldsymbol{z}_0 = \boldsymbol{x}$, $\boldsymbol{z} = \boldsymbol{z}_{|S_2|}$ and $\boldsymbol{z}_s$ replaces $x_{j_s}(j_s \in S_2)$ in $\boldsymbol{z}_{s-1}$, to gradually transform $\boldsymbol{x}$ to a fully continuous vector $\boldsymbol{z}$. As a result, we have

$$W(F(\boldsymbol{z}), F(\boldsymbol{x})) \leq \sum_{s=1}^{|S_2|} W(F(\boldsymbol{z}_s), F(\boldsymbol{z}_{s-1})) \leq |S_2|\sqrt{\frac{2\eta^2}{\pi}}\varepsilon.$$

Let $\tilde{\mathcal{T}} = F(\boldsymbol{z})$ and denote the mixed distribution $F(\boldsymbol{x})$ by $\mathcal{T}$. Then, by the triangle inequality, it is easy to see that for the generative model G', we have

$$W_1(G_\sharp\rho, \mathcal{T}) \leq W_1(G_\sharp\rho, \tilde{\mathcal{T}}) + W_1(\tilde{\mathcal{T}}, \mathcal{T}) < (1 + |S_2|\sqrt{\frac{2\eta^2}{\pi}})\varepsilon \leq (1 + \sqrt{\frac{2}{\pi}}d\eta)\varepsilon.$$

□

2.7.1.2 Proof of Theorem 2

With a little abuse of the notation, we use $y(\boldsymbol{x}, \boldsymbol{z})$ to denote the response to task $\boldsymbol{x}$ given by the virtual participant with profile $\boldsymbol{z}$. When it won't cause confusion, we further simplify the notation and write it as $y(\boldsymbol{z})$. Before we go ahead to prove Theorem 2, we introduce the following lemma on ambiguity decomposition [239, 215]:

Lemma 2.7.1. *Given the "ground truth" $\overline{y}$, and the noisy responses of a population $\tilde{y}_i(i = 1, \cdots, N)$ with their average $\overline{\tilde{y}} = \frac{1}{N}\sum_{i=1}^N \tilde{y}_i$, we have the following ambiguity decomposition:*

$$\ell(\overline{y}, \overline{\tilde{y}}) = \frac{1}{N}\sum_{i=1}^N \ell(\overline{y}, \tilde{y}_i) - \frac{1}{N}\sum_{i=1}^N \ell(\overline{\tilde{y}}, \tilde{y}_i).$$

Proof. We first notice that the squared loss admits the following bias-variance decomposition:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{z}\sim\mathcal{T}}[\ell(\overline{y}, \tilde{y}(\boldsymbol{z}))] = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{T}}\Big[&\ell\big(\overline{y}, \mathbb{E}_{\boldsymbol{z}\sim\mathcal{T}}[\tilde{y}(\boldsymbol{z})]\big) + \ell\big(\mathbb{E}_{\boldsymbol{z}\sim\mathcal{T}}[\tilde{y}(\boldsymbol{z})], \tilde{y}(\boldsymbol{z})\big) \\ &+ 2\big(\overline{y} - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{T}}[\tilde{y}(\boldsymbol{z})]\big)\big(\mathbb{E}_{\boldsymbol{z}\sim\mathcal{T}}[\tilde{y}(\boldsymbol{z})] - \tilde{y}(\boldsymbol{z})\big)\Big] \\ = \ell\big(\overline{y}, \mathbb{E}_{\boldsymbol{z}\sim\mathcal{T}}[\tilde{y}(\boldsymbol{z})]\big) &+ \mathbb{E}_{\mathcal{T}}\Big[\ell\big(\mathbb{E}_{\boldsymbol{z}\sim\mathcal{T}}[\tilde{y}(\boldsymbol{z})], \tilde{y}(\boldsymbol{z})\big)\Big].\end{aligned}$$

By adopting $\mathcal{T}$ as a finite sample population $U = \{u_i\}_{i=1}^N$, we can rewrite the decomposition as

$$\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},\tilde{y}_i) = \ell\big(\overline{y},\overline{\tilde{y}}\big) + \frac{1}{N}\sum_{i=1}^{N}\ell\big(\overline{\tilde{y}},\tilde{y}_i\big).$$

A simple rearrangement of the terms completes the proof. □

With this ambiguity decomposition, by taking the expected risk of $\overline{\tilde{y}}$ we have

$$\begin{aligned}
L &= \mathbb{E}_{\mathcal{T}}\Big[\mathbb{E}_{\mathcal{D}}\big[\mathbb{E}_{\boldsymbol{x}\sim\mathcal{X},y\sim\mathcal{Y}}[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},\tilde{y}_i)]\big]\Big] - \mathbb{E}_{\mathcal{T}}\Big[\mathbb{E}_{\mathcal{D}}\big[\mathbb{E}_{\boldsymbol{x}\sim\mathcal{X},y\sim\mathcal{Y}}[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{\tilde{y}},\tilde{y}_i)]\big]\Big] \\
&= \mathbb{E}_{\mathcal{X}}\Big[\mathbb{E}_{\mathcal{T},\mathcal{D}}\big[\mathbb{E}_{y\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},\tilde{y}_i)]\big]\Big] - \mathbb{E}_{\mathcal{T},\mathcal{D}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{\tilde{y}},\tilde{y}_i)\big]\Big].
\end{aligned}$$

Now we notice that the first term has the following decomposition:

$$\begin{aligned}
&\mathbb{E}_{\mathcal{T},\mathcal{D}}\Big[\mathbb{E}_{y,\tilde{y}\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},\tilde{y}_i)\big]\Big] \\
=&\mathbb{E}_{\mathcal{T},\mathcal{D}}\Big[\mathbb{E}_{y,\tilde{y}\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},\overline{\tilde{y}}_i)\big]\Big] \\
&+\mathbb{E}_{\mathcal{T},\mathcal{D}}\Big[\mathbb{E}_{\tilde{y}\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{\tilde{y}}_i,\tilde{y}_i)\big]\Big] \\
=&\mathbb{E}_{\mathcal{T},\mathcal{D}}\Big[\mathbb{E}_{y,\tilde{y}\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},\overline{\tilde{y}}_i)\big]\Big] + \mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\tilde{\eta}_i^2\big] \\
=&\mathbb{E}_{\mathcal{T},\mathcal{D}}\Big[\mathbb{E}_{y\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},\overline{y}_i)\big]\Big] \\
&+\mathbb{E}_{\mathcal{T},\mathcal{D}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i,\overline{\tilde{y}}_i)\big] + \mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\tilde{\eta}_i^2\big].
\end{aligned}$$

This is followed by

$$\begin{aligned}
&\mathbb{E}_{\mathcal{T},\mathcal{D}}\Big[\mathbb{E}_{y\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},\overline{y}_i)\big]\Big] \\
=&\mathbb{E}_{\mathcal{T},\mathcal{D}}\Big[\mathbb{E}_{y\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y},y_i)\big]\Big]
\end{aligned}$$

$$
\begin{aligned}
&+ \mathbb{E}_{\mathcal{T},\mathcal{D}}\Big[\mathbb{E}_{y\sim\mathcal{Y}|\mathcal{X},\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(y_i,\overline{y}_i)\big]\Big] \\
=&\mathbb{E}_{\substack{\boldsymbol{v}_1,\ldots,\boldsymbol{v}_N\\ \sim\mathcal{T}}}\Big[\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{y_i\sim\mathcal{Y}|\mathcal{X},\boldsymbol{v}_i}[\ell(\overline{y},y_i)]\Big] \\
&+ \mathbb{E}_{\mathcal{T}}\Big[\frac{1}{N}\sum_{i=1}^{N}\eta_i^2\Big].
\end{aligned}
$$

Putting all together, we can obtain the full decomposition:

$$
\begin{aligned}
L = \mathbb{E}_{\mathcal{X}}\Big[&\mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\eta_i^2\big] \\
&+ \mathbb{E}_{\substack{\boldsymbol{v}_1,\ldots,\boldsymbol{v}_N\\ \sim\mathcal{T}}}\Big[\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{y_i\sim\mathcal{Y}|\mathcal{X},\boldsymbol{v}_i}[\ell(\overline{y},y_i)]\Big] \\
&+ \mathbb{E}_{\mathcal{T},\mathcal{D}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i,\overline{\tilde{y}}_i)\big] + \mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\tilde{\eta}_i^2\big] \\
&- \mathbb{E}_{\mathcal{T},\mathcal{D}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{\tilde{y}},\tilde{y}_i)\big]\Big].
\end{aligned}
$$

2.7.1.3 Proof of Theorem 3

By theorem 2, we have

$$
L = L_1 + L_2 + \mathbb{E}_{\mathcal{T},\mathcal{D}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i,\overline{\tilde{y}}_i)\big] + \mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\tilde{\eta}_i^2\big] - E_{\mathcal{T},\mathcal{D}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{\tilde{y}},\tilde{y}_i)\big]\Big].
$$

while Proposition 3 gives us

$$
L' = L_1 + L_2 + \mathbb{E}_{\mathcal{T}}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i,y_{ref})\big] + \eta(t).
$$

Therefore, we have

$$
\begin{aligned}
L - L' = \mathbb{E}\big[&\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i,\overline{\tilde{y}}_i)\big] + \mathbb{E}\big[\frac{1}{N}\sum_{i=1}^{N}\tilde{\eta}_i^2\big] \\
&- \mathbb{E}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{\tilde{y}},\tilde{y}_i)\big] - \mathbb{E}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i,y_{ref})\big] - \eta(t).
\end{aligned}
$$

By Eq. (8), we have $\tilde{y}_i = y_{ref} + \delta_i + \tilde{\varepsilon}_i$ where, for a specific $\boldsymbol{x}$, $y_{ref} = \Phi(\boldsymbol{x})$, $\delta_i = \delta(\boldsymbol{x}, \boldsymbol{v}_i)$, $\mathbb{E}[\tilde{\varepsilon}_i] = 0$ and $\mathbb{E}[\tilde{\varepsilon}_i^2] = \tilde{\eta}^2$. It further gives $\overline{\tilde{y}}_i = y_{ref} + \frac{1}{N}\sum_{i=1}^{N}\delta_i$. Thus, we can obtain

$$
\begin{aligned}
L - L' = &\mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i, y_{ref} + \frac{1}{N}\sum_{i=1}^{N}\delta_i)\Big] + \mathbb{E}\big[\frac{1}{N}\sum_{i=1}^{N}\tilde{\eta}_i^2\big] \\
&- \mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^{N}\ell(y_{ref} + \delta_i + \tilde{\varepsilon}_i, y_{ref} + \frac{1}{N}\sum_{i=1}^{N}\delta_i)\Big] \\
&- \mathbb{E}\big[\frac{1}{N}\sum_{i=1}^{N}\ell(\overline{y}_i, y_{ref})\big] - \eta(t) \\
=&\mathbb{E}\Big[(\frac{1}{N}\sum_{i=1}^{N}\delta_i)^2\Big] - 2\mathbb{E}\Big[(\frac{1}{N}\sum_{i=1}^{N}\delta_i)(\frac{1}{N}\sum_{i=1}^{N}\overline{y}_i - y_{ref})\Big] \\
&- \mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^{N}(\delta_i - \frac{1}{N}\sum_{i=1}^{N}\delta_i)^2\Big] - \eta(t) \\
=&\mathbb{E}\Big[(\frac{1}{N}\sum_{i=1}^{N}\delta_i)^2\Big] - 2\mathbb{E}\Big[(\frac{1}{N}\sum_{i=1}^{N}\delta_i)(y^{**} - y_{ref})\Big] \\
&- \mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^{N}(\delta_i - \frac{1}{N}\sum_{i=1}^{N}\delta_i)^2\Big] - \eta(t) \\
=&2\mathbb{E}\Big[(\frac{1}{N}\sum_{i=1}^{N}\delta_i)^2\Big] - 2\mathbb{E}\Big[(\frac{1}{N}\sum_{i=1}^{N}\delta_i)(y^{**} - y_{ref})\Big] \\
&- \mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^{N}\delta_i^2\Big] - \eta(t) \\
=&\mathbb{E}[2P^2 - 2P(y^{**} - y_{ref}) - Q - \eta(t)]
\end{aligned}
$$

where y^{**} is the gold-standard response of the real human population that is not affected by the digital population, $P = \frac{1}{N}\sum_{i=1}^{N}\delta_i$,and $Q = \frac{1}{N}\sum_{i=1}^{N}\delta_i^2$. Now let $\Delta = y^{**} - y_{ref}$. Noting that $\mathbb{E}[P^2] = \mathbb{E}^2[P] + Var[P] = \mu_\delta^2 + \frac{1}{N}\varepsilon_\delta^2 - \frac{1}{N^2}\sum_{i=1}^{N}\mathbb{E}^2[\delta_i]$, $\mathbb{E}[P] = \mu_\delta$, and $\mathbb{E}[Q] = \varepsilon_\delta^2$, it yields

$$
\begin{aligned}
L - L' &= 2\mu_\delta^2 + \frac{2}{N}\varepsilon_\delta^2 - \frac{2}{N^2}\sum_{i=1}^{N}\mathbb{E}^2[\delta_i] - 2\mu_\delta(y^{**} - y_{ref}) - \varepsilon_\delta^2 - \eta(t) \\
&\leq 2(1 - \frac{1}{N})\mu_\delta^2 - 2\mu_\delta\Delta - (1 - \frac{2}{N})\varepsilon_\delta^2 - \eta(t)
\end{aligned}
$$

$$= \mathbb{E}\Big[\frac{2(N-1)}{N}[\mu_\delta - \frac{N}{2(N-1)}\Delta]^2 - \frac{N}{2(N-1)}\Delta^2 - (1-\frac{2}{N})\varepsilon_\delta^2 - \eta(t)\Big],$$

For brevity, we let $A = \frac{N}{2(N-1)}, B = 1 - \frac{2}{N}, C = \frac{N-2}{2(N-1)} = 1 - A$. Obviously, when $N \geq 2$, $0 < A \leq 1$ and $B, C \geq 0$. The equation above gives a sufficient condition of $L - L' \leq 0$:

$$A\Delta - \sqrt{A^2\Delta^2 + AB\varepsilon_\delta^2 + A\eta(t)} \leq \mu_\delta \leq A\Delta + \sqrt{A^2\Delta^2 + AB\varepsilon_\delta^2 + A\eta(t)}$$

It is equivalent to

$$\Delta - \sqrt{A^2\Delta^2 + AB\varepsilon_\delta^2 + A\eta(t)} - C\Delta \leq \mu_\delta \leq \Delta + \sqrt{A^2\Delta^2 + AB\varepsilon_\delta^2 + A\eta(t)} - C\Delta$$

Let $\Delta_L = \sqrt{A^2\Delta^2 + AB\varepsilon_\delta^2 + A\eta(t)} + C\Delta$ and $\Delta_U = \sqrt{A^2\Delta^2 + AB\varepsilon_\delta^2 + A\eta(t)} - C\Delta$. Meanwhile, we notice the derivatives are:

$$\Delta_L' = \frac{A^2\Delta}{\sqrt{A^2\Delta^2 + AB\varepsilon_\delta^2 + A\eta(t)}} + C$$
$$\Delta_U' = \frac{A^2\Delta}{\sqrt{A^2\Delta^2 + AB\varepsilon_\delta^2 + A\eta(t)}} - C$$

Let us consider

$$\Delta_0 = \sqrt{\frac{C^2[B\varepsilon_\delta^2 + \eta(t)]}{A(A^2 - C^2)}} = \frac{N-2}{N}\sqrt{\frac{(N-2)\varepsilon_\delta^2 + N\eta(t)}{2}}$$

The stationary points lie at $\Delta = -\Delta_0$ and $\Delta = \Delta_0$, respectively. Here, it is easy to see $A > C \geq 0$ for $N \geq 2$, thus $\Delta_0 = \frac{C^2[B\varepsilon_\delta^2+\eta(t)]}{A(A^2-C^2)} \geq 0$. Therefore, we have the following observations: When $\Delta > 0$, as Δ increases, Δ_L will increase, whereas Δ_U will decrease when $\Delta < \Delta_0$ and become increasing for $\Delta \geq \Delta_0$. When $\Delta < 0$, as Δ increases, Δ_L will decrease until Δ reaches $-\Delta_0$ and become increasing, whereas Δ_U will decrease. Thus, we can find the minimum of the bounds:

$$\Delta_L, \Delta_U \geq \frac{1}{N}\sqrt{2[(N-2)\varepsilon_\delta^2 + N\eta(t)]}$$

As a result, we have two cases of the bounds depending on N and can build confidence interval as follows:

- N **is sufficiently large**. When $\Delta_0 = \frac{N-2}{N}\sqrt{\frac{(N-2)\varepsilon_\delta^2 + N\eta(t)}{2}} \geq \kappa_\alpha$, Δ_L and Δ_U will be monotonically increasing and decreasing, within $|\Delta| \leq \kappa_\alpha$. In this case, we let

$$h_\Delta(\kappa_\alpha) = \frac{\sqrt{N^2\kappa_\alpha^2 + 2(N-1)(N-2)\varepsilon_\delta^2 + 2N(N-1)\eta(t)} - (N-2)\kappa_\alpha}{2(N-1)}$$

- N **is small**. Given $\Delta_0 = \frac{N-2}{N}\sqrt{\frac{(N-2)\varepsilon_\delta^2 + N\eta(t)}{2}} < \kappa_\alpha$, we let

$$h_\Delta(\kappa_\alpha) = \frac{1}{N}\sqrt{2[(N-2)\varepsilon_\delta^2 + N\eta(t)]}$$

We can build the interval as

$$B_\alpha(\Delta) = [\Delta - h_\Delta(C), \Delta + h_\Delta(C)]$$

Since $|\Delta| < \kappa_\alpha$ with at least probability $1-\alpha$, if $\mu_\delta \in B_\alpha(\Delta)$, with the same probability, we can guarantee $L \leq L'$.

2.7.1.4 Proof of Theorem 4

This proof follows [216]. Let $\mathcal{E}_1 = \{\mathbb{E}_{\boldsymbol{v}\sim\mathcal{T}}[f(\boldsymbol{x},\boldsymbol{v}) - y] \in B_\gamma^1(\theta^*)\}$ and $\mathcal{E}_2 = \{\mathbb{E}_{\boldsymbol{v}\sim\mathcal{T}}[\theta^* - f(\boldsymbol{x},\boldsymbol{v})] \in B_{\alpha-\gamma}^2(\theta^*)\}$. From the conditions, we have $P(\mathcal{E}_1) \geq 1-\gamma$ and $P(\mathcal{E}_2) \geq 1-(\alpha-\gamma)$. Consider the event $\mathcal{E} = \mathcal{E}_1 \bigcap \mathcal{E}_2$. It is easy to see $P(\mathcal{E}) = 1 - P(\mathcal{E}_1^c \bigcup \mathcal{E}_2^c) \geq 1 - P(\mathcal{E}_1^c) - P(\mathcal{E}_2^c) = P(\mathcal{E}_1) + P(\mathcal{E}_2) - 1 \geq 1 - [\gamma + (\alpha-\gamma)] = 1-\alpha$. On the event $\mathcal{E}$, we have

$$\begin{aligned}
\mathbb{E}[\theta^* - y|\boldsymbol{x}] &= \mathbb{E}[\theta^* - y|\boldsymbol{x}] - \mathbb{E}[\theta^* - f(\boldsymbol{x},\boldsymbol{v})|\boldsymbol{x}] + \mathbb{E}[\theta^* - f(\boldsymbol{x},\boldsymbol{v})|\boldsymbol{x}] \\
&= \mathbb{E}[f(\boldsymbol{x},\boldsymbol{v}) - y|\boldsymbol{x}] + \mathbb{E}[\theta^* - f(\boldsymbol{x},\boldsymbol{v})|\boldsymbol{x}] \\
&\in B_\gamma^1(\theta^*) + B_{\alpha-\gamma}^2(\theta^*).
\end{aligned}$$

Noticing $\theta^* = \mathbb{E}[y|\boldsymbol{x}]$, we have $\mathbb{E}[\theta^* - y|\boldsymbol{x}] = 0$. Thus we have $0 \in B_\gamma^1(\theta^*) + B_{\alpha-\gamma}^2(\theta^*)$, which turns to be a necessary condition. This completes the proof.

2.7.1.5 Proof of Theorem 5

We borrow the techniques from [216] and show that $y^* \notin B_\alpha^y$ with probability at most α when $n, N \to \infty$. First, we denote $\theta = y^*$ and notice that

$$\begin{aligned}\theta - \overline{\tilde{y}} &= \theta - \frac{1}{N}\sum_{i=1}^{N}\tilde{y}_i = \theta - \frac{1}{N}\sum_{i=1}^{N}[\Phi^{(i)}(\boldsymbol{x}) + \delta_i] \\ &= \Big(\mathbb{E}[\Phi(\boldsymbol{x})] - \frac{1}{N}\sum_{i=1}^{N}\Phi^{(i)}(\boldsymbol{x})\Big) + \Big(\theta - \mathbb{E}[\Phi(\boldsymbol{x})] - \frac{1}{N}\sum_{i=1}^{N}\delta_i\Big).\end{aligned}$$

Let $r_i' = -r_i$, $\Delta_\delta = \theta - \mathbb{E}[\Phi(\boldsymbol{x})] - \overline{\delta}$ and $\Delta_\Phi = \mathbb{E}[\Phi(\boldsymbol{x})] - \frac{1}{N}\sum_{i=1}^{N}\Phi^{(i)}(\boldsymbol{x})$. Thus, we have $\theta - \overline{\tilde{y}} = \Delta_\delta + \Delta_\Phi$. By central limit theorem, we have

$$\begin{aligned}&\sqrt{n}(\overline{r}' - \mathbb{E}[\overline{r}']) \xrightarrow{d} \mathcal{N}(0, \sigma_r^2), \\ &\sqrt{N}(\Delta_\delta - \mathbb{E}[\Delta_\delta]) \xrightarrow{d} \mathcal{N}(0, \sigma_\delta^2), \\ &\sqrt{N}(\Delta_\Phi - \mathbb{E}[\Delta_\Phi]) \xrightarrow{d} \mathcal{N}(0, \eta_\Phi(t)).\end{aligned}$$

Thus, we have

$$\begin{aligned}&\sqrt{N}(\Delta_\delta + \Delta_\Phi + \overline{r}' - \mathbb{E}[\Delta_\delta + \Delta_\Phi + \overline{r}']) \\ =&\sqrt{n}\cdot\sqrt{\frac{N}{n}}(\overline{r}' - \mathbb{E}[\overline{r}']) + \sqrt{N}\Big\{(\Delta_\delta - \mathbb{E}[\Delta_\delta]) + (\Delta_\Phi - \mathbb{E}[\Delta_\delta])\Big\} \\ \to&\mathcal{N}(0, \frac{1}{p}\sigma_r^2 + \sigma_\delta^2 + \eta(t)).\end{aligned}$$

Let $\hat{\sigma}^2 = \frac{1}{p}\hat{\sigma}_r^2 + \hat{\sigma}_\delta^2 + \eta(t) = \frac{N}{n}\hat{\sigma}_r^2 + \hat{\sigma}_\delta^2 + \eta(t)$. It is easy to see that this is a consistent estimate of the variance $\frac{1}{p}\sigma_r^2 + \sigma_\delta^2 + \eta(t)$. Therefore, we have

$$\lim_{n,N\to\infty} P(|(\Delta_\delta + \Delta_\Phi + \overline{r}') - \mathbb{E}[\Delta_\delta + \Delta_\Phi + \overline{r}']| \geq z_{1-\frac{\alpha}{2}}\frac{\hat{\sigma}}{\sqrt{N}}) \leq \alpha.$$

Since we know

$$\Delta_\delta + \Delta_\Phi + \overline{r}' = \Delta_\delta + \Delta_\Phi - \overline{r} = \theta - \overline{\tilde{y}} + \overline{\tilde{y}} - \overline{y} = \theta - \overline{y},$$

we can easily obtain

$$\mathbb{E}[\Delta_\delta + \Delta_\Phi + \overline{r}'] = \mathbb{E}[\theta - \overline{y}] = 0.$$

Thus, we have

$$\lim_{n,N\to\infty} P(|\Delta_\delta + \Delta_\Phi + \overline{r}'| \geq z_{1-\frac{\alpha}{2}} \frac{\hat{\sigma}}{\sqrt{N}}) \leq \alpha,$$

which is equivalent to

$$\lim_{n,N\to\infty} P(|\theta - \overline{\tilde{y}} - \overline{r}| \geq z_{1-\frac{\alpha}{2}} \sqrt{\frac{\eta(t)}{N} + \frac{\sigma_\delta^2}{N} + \frac{\sigma_r^2}{n}}) \leq \alpha.$$

This results in

$$\lim_{n,N\to\infty} P(|\theta - \overline{\tilde{y}}| \geq |\overline{r}| + z_{1-\frac{\alpha}{2}} \sqrt{\frac{\eta(t)}{N} + \frac{\sigma_\delta^2}{N} + \frac{\sigma_r^2}{n}}) \leq \alpha.$$

Noticing that

$$|\overline{r}|^2 = |\frac{1}{n}\sum_{i=1}^{n} y_i - \frac{1}{n}\sum_{i=1}^{n} \tilde{y}_i|^2 \leq \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \leq \varepsilon_0^2,$$

which gives $|\overline{r}| \leq \varepsilon_0$, we get

$$\lim_{n,N\to\infty} P(|\theta - \overline{\tilde{y}}| \geq \varepsilon_0 + z_{1-\frac{\alpha}{2}} \sqrt{\frac{\eta(t)}{N} + \frac{\sigma_\delta^2}{N} + \frac{\sigma_r^2}{n}}) \leq \alpha.$$

2.7.2 Example of Prompts

2.7.2.1 Zero-shot prompt

By default, for all the LLM-based decision-making methods, we use zero-shot prompts which provides LLM with the problem description $\boldsymbol{x}_t$, the specific requirements $\mathcal{R}_t$ on the decisions and the context $\mathcal{C}_t$. $\boldsymbol{x}_t$ describes the specific problem the workers need to make decisions on. $\mathcal{R}_t$ describes what kind of decisions need to be made, e.g., if the decisions should be a score, the scale of the score should be included in $\mathcal{R}_t$. And $\mathcal{C}_t$ encodes information about the decision-making scenario and the decision-maker LLM should simulate. An example of zero-shot prompt is given below in Figure 2.10.

2.7.2.2 Multi-Persona prompt

With the idea of using multiple personas in prompting for self-collaboration [232, 229], we use LLM to simulate different individuals by building multiple personas with additional context

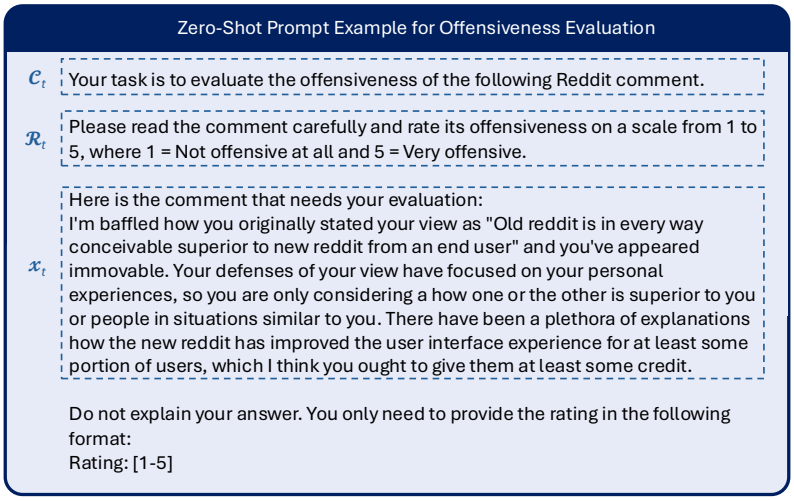


Fig. 2.10: Example of zero-shot prompt for offensiveness evaluation.

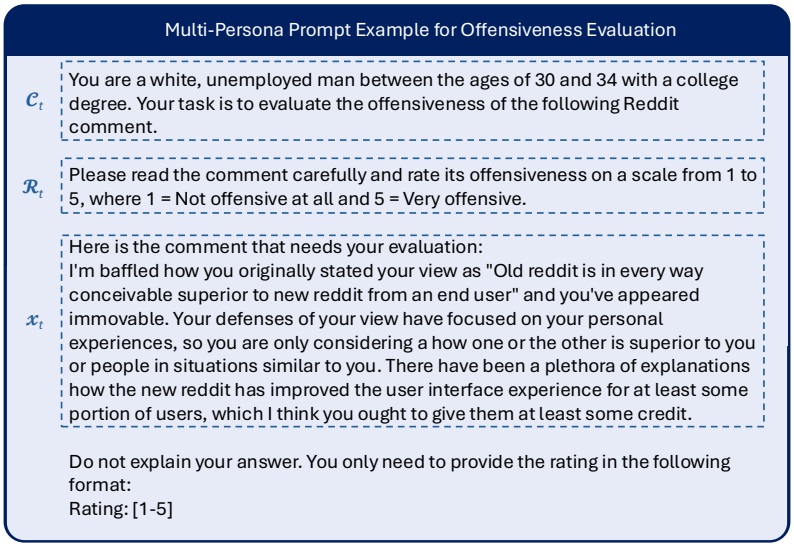


Fig. 2.11: Example of multi-persona prompt for offensiveness evaluation.

on their profiles. We use similar prompts as zero-shot prompting for making decisions but only change the context $\mathcal{C}_t$ to assign the persona. An example of multi-persona prompt is provided below in Figure 2.11.

2.7.2.3 Self-Consistency (SC) prompt

Following [230, 231], we perform self-consistency prompting by taking the majority vote of multiple decisions generated by LLM with temperature 0.5. For the generation of each decision, we use the same prompt as zero-shot prompting.

2.7.3 Additional Experimental Results

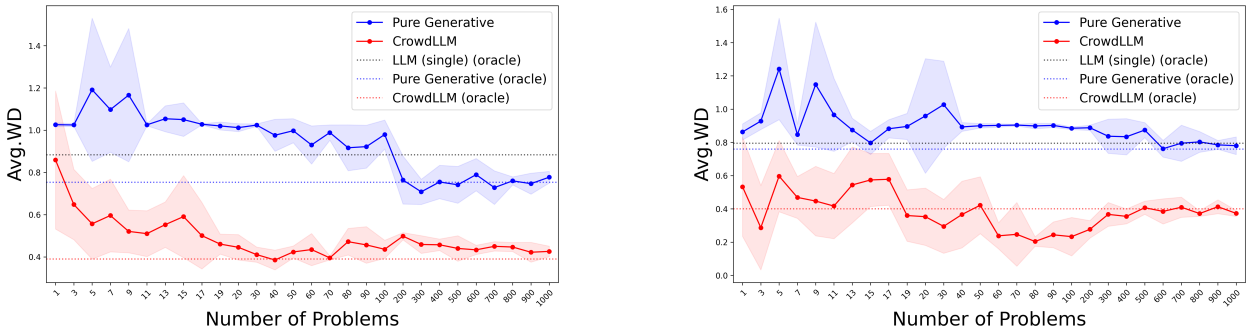
Section C.1 discusses the impact of different LLM backbones on the performance. **Section C.2-6** provides experimental results in addition to **Section 5** of the main text.

2.7.3.1 The impact of the number of problems

This subsection examines how the total number of problems affects model performance (MAE, RMSE, Avg. WD) for Pure Generative Model and our proposed CrowdLLM on the *Offensiveness* and *QA Difficulty* datasets. As Figures 2.12-2.14 demonstrates, CrowdLLM consistently outperforms Pure Generative with lower errors across all datasets as problem numbers increase. Both models improve as problems grow from few to moderate (e.g., 1 to 100-200), after which gains diminish and metrics stabilize. Notably, CrowdLLM achieves better absolute errors and reaches stable, high-quality performance with fewer problems than Pure Generative (e.g., on *Offensiveness* and *QA Difficulty*, CrowdLLM stabilizes around 20-100 problems, while Pure Generative improves more gradually). This highlights CrowdLLM's data efficiency.

2.7.3.2 The impact of the number of unique workers

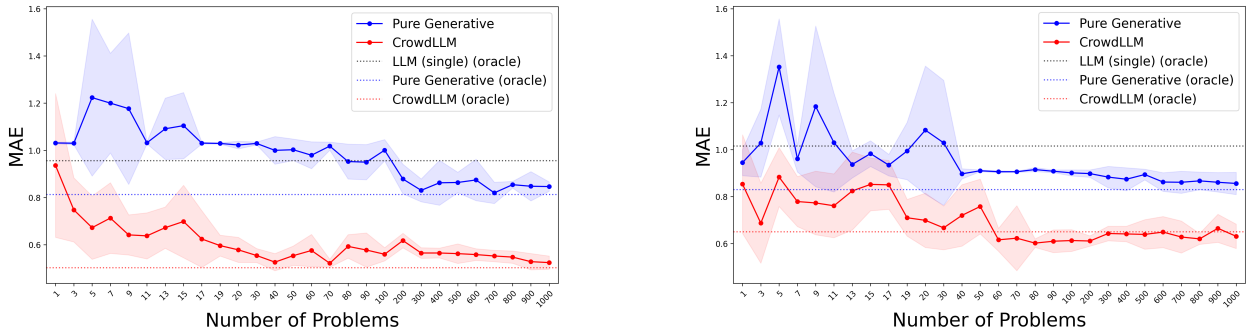
This section analyzes how the number of unique workers affects Pure Generative Model and CrowdLLM performance (MAE, RMSE, Avg. WD) on the *Offensiveness* and *QA Difficulty* datasets. As Figures 2.15-2.17 shows, CrowdLLM consistently shows lower error rates than



(a) Offensiveness dataset

(b) QA Difficulty dataset

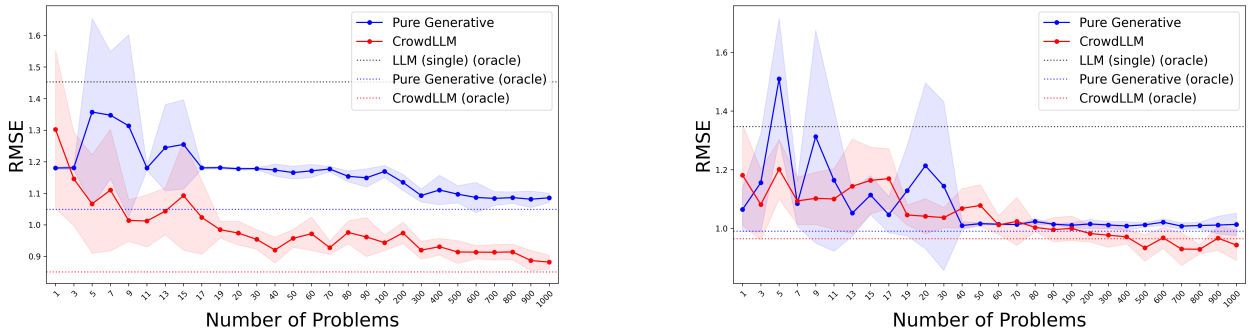
Fig. 2.12: Avg. WD vs. Number of Problems.



(a) Offensiveness dataset

(b) QA Difficulty dataset

Fig. 2.13: MAE vs. Number of Problems.



(a) Offensiveness dataset

(b) QA Difficulty dataset

Fig. 2.14: RMSE vs. Number of Problems.

Pure Generative across all metrics, regardless of unique worker count. For both models, errors substantially decrease when unique workers increase from few (e.g., 1) to moderate (e.g., 20-50), as diverse perspectives improve data quality. Beyond a point (e.g., 50-100 workers for CrowdLLM, potentially more for Pure Generative), benefits diminish and metrics stabilize. CrowdLLM generally reaches a better performance plateau with fewer unique workers.

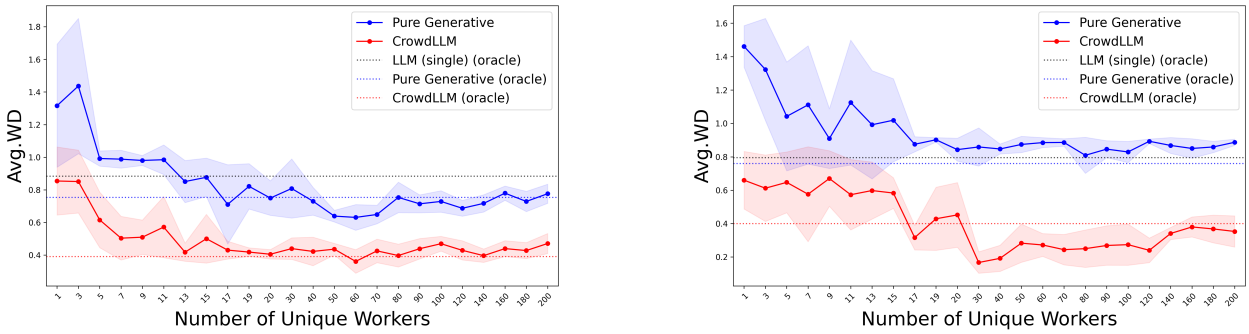
2.7.3.3 The impact of the number of ratings

This section assesses how total ratings impact Pure Generative and CrowdLLM performance (MAE, RMSE, Avg. WD) on the *Offensiveness*, and *QA Difficulty* datasets. From Figures 2.18-2.20, we see that CrowdLLM consistently outperforms Pure Generative with lower errors as ratings increase. Both models improve with more ratings, with significant gains when increasing from few (e.g., 1-100) to hundreds/thousands (e.g., 1000-2000). Beyond a high volume (e.g., >2000-5000), improvements slow down and performance stabilizes. CrowdLLM achieves superior performance and often reaches optimal levels with fewer total ratings than Pure Generative, underscoring its data efficiency.

2.7.3.4 The impact of the number of workers per problem

This section analyzes how workers per problem (ranging from 1 to 8) influence Pure Generative and CrowdLLM performance (MAE, RMSE, Avg. WD) across the three datasets. The results are shown in Figures 2.21-2.23.

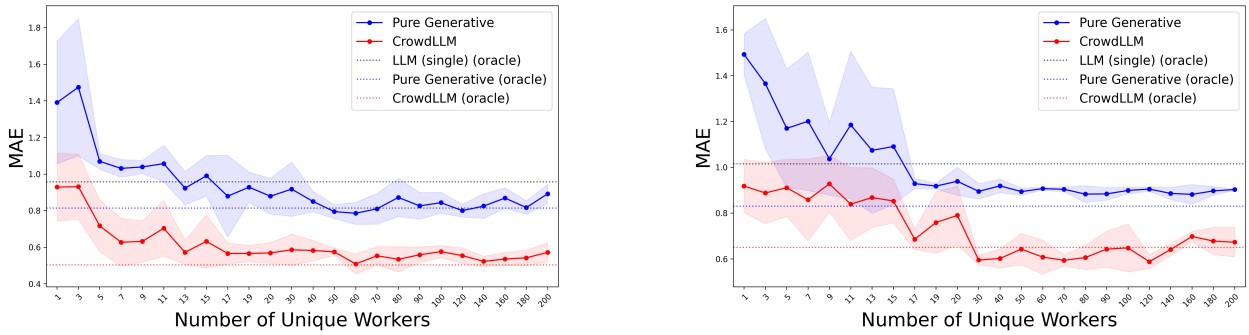
CrowdLLM consistently shows markedly lower errors than Pure Generative. Increasing workers per problem from one generally improves aggregated label quality and reduces errors for both models, most notably when moving from 1 to 2-3 workers. CrowdLLM typically reaches optimal performance or diminishing returns with few workers (e.g., 2-4); for instance, on *Offensiveness*, its Avg. WD stabilizes after 3 workers. Adding more workers (up to 8) offers little further benefit for CrowdLLM. Pure Generative also improves but maintains higher error rates than CrowdLLM.



(a) Offensiveness dataset

(b) QA Difficulty dataset

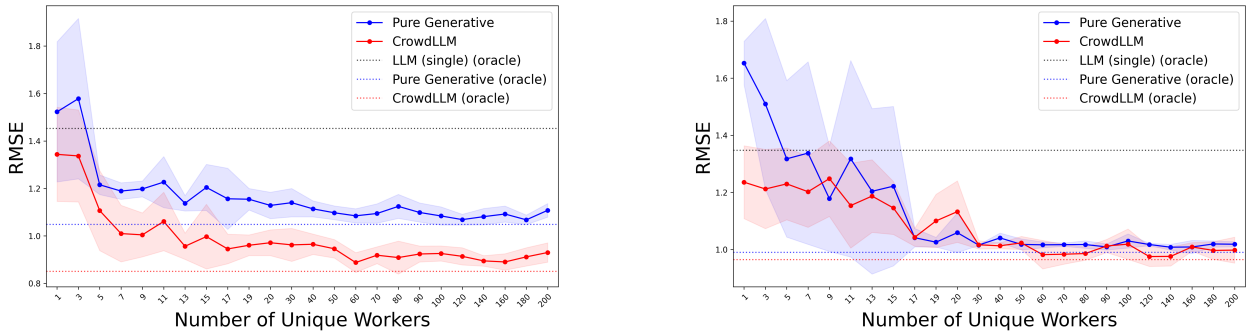
Fig. 2.15: Avg. WD vs. Number of Unique Workers.



(a) Offensiveness dataset

(b) QA Difficulty dataset

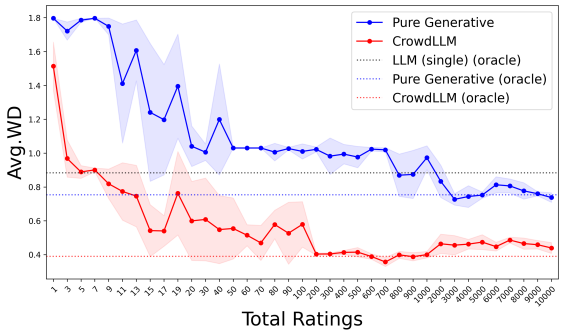
Fig. 2.16: MAE vs. Number of Unique Workers.



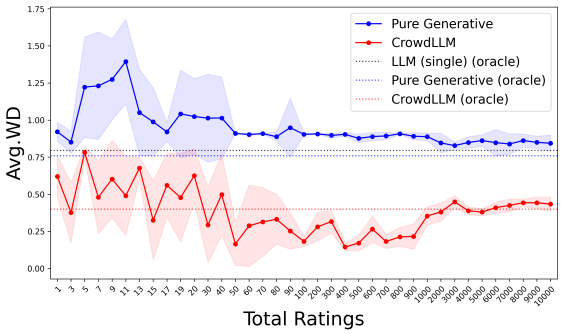
(a) Offensiveness dataset

(b) QA Difficulty dataset

Fig. 2.17: RMSE vs. Number of Unique Workers.

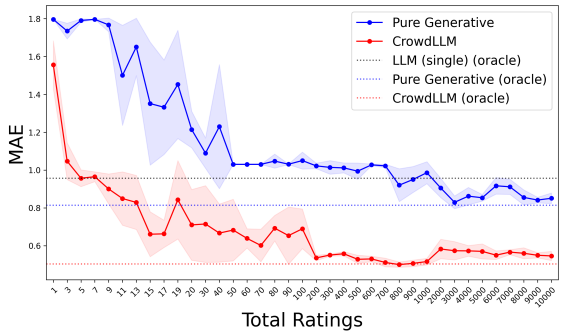


(a) Offensiveness dataset

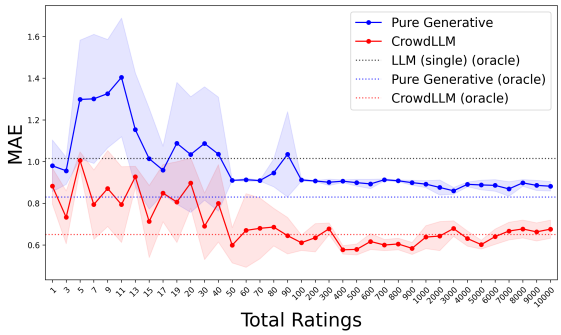


(b) QA Difficulty dataset

Fig. 2.18: Avg. WD vs. Number of Ratings.

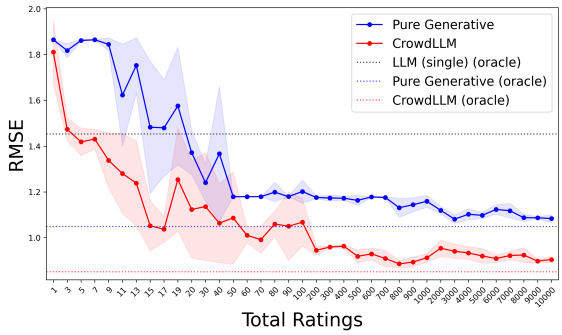


(a) Offensiveness dataset

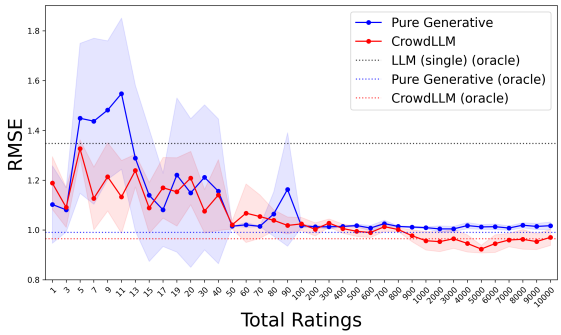


(b) QA Difficulty dataset

Fig. 2.19: MAE vs. Number of Ratings.

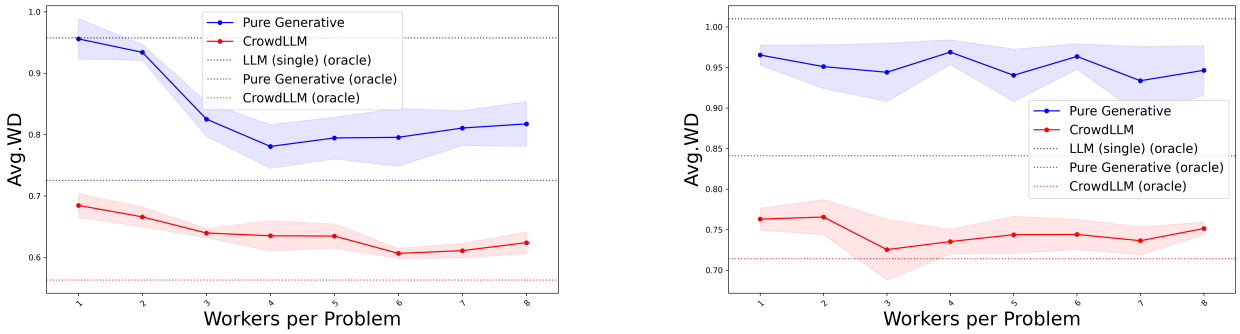


(a) Offensiveness dataset



(b) QA Difficulty dataset

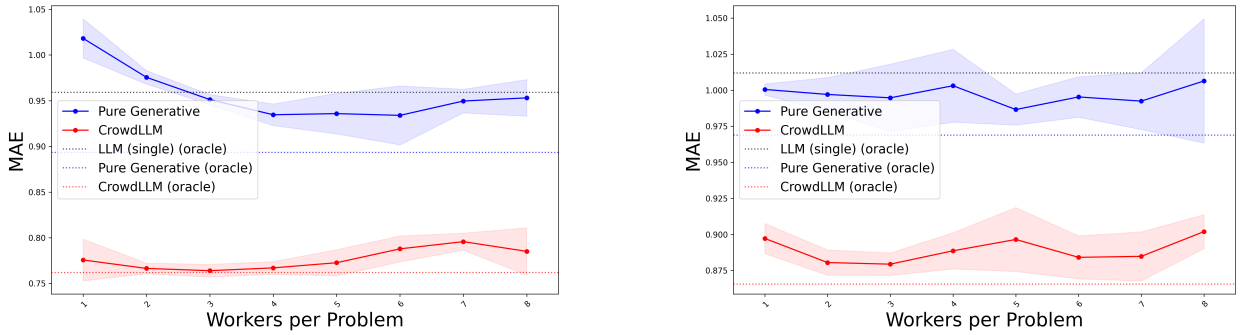
Fig. 2.20: RMSE vs. Number of Ratings.



(a) Offensiveness dataset

(b) QA Difficulty dataset

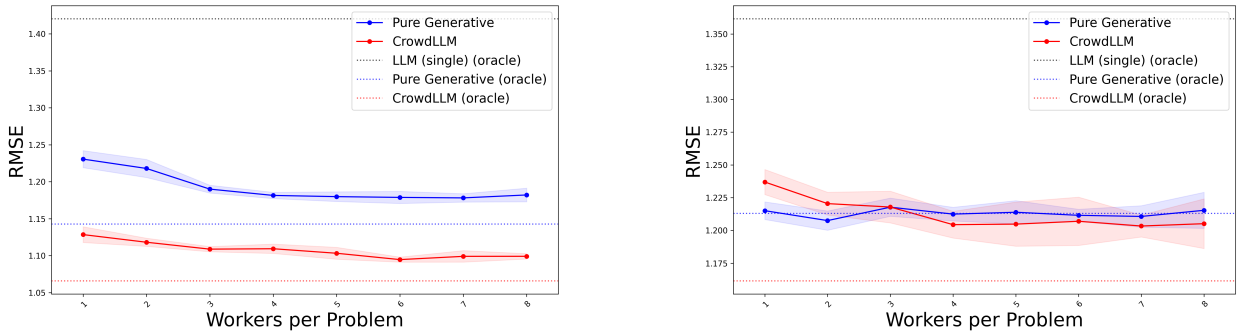
Fig. 2.21: Avg. WD vs. Number of Workers per Problem.



(a) Offensivenss dataset

(b) QA Difficulty dataset

Fig. 2.22: MAE vs. Number of Workers per Problem.



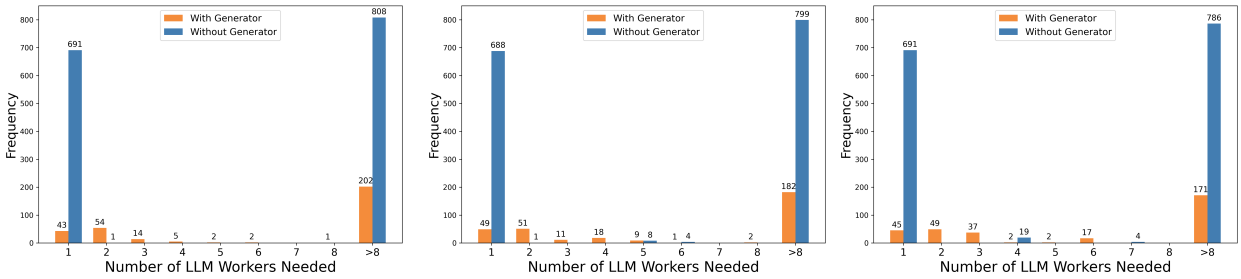
(a) Offensiveness dataset

(b) QA Difficulty dataset

Fig. 2.23: RMSE vs. Number of Workers per Problem.

2.7.3.5 Cost-saving prospect of CrowdLLM

To assess CrowdLLM's cost-effectiveness, we analyze the LLMs needed to approximate ground-truth ratings within various tolerances ($\pm\{0.1, 0.2, 0.3, 0.4\}$) across datasets.

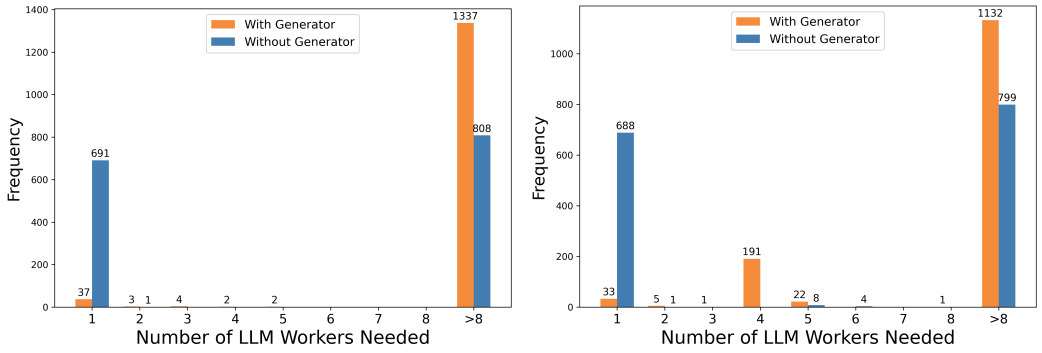


(a) Ground truth rating ± 0.1 (b) Ground truth rating ± 0.2 (c) Ground truth rating ± 0.3

Fig. 2.24: Number of LLM workers needed to reach different performance ranges with and without generator-based rating on Offensiveness dataset. The generator is trained on more than 13,000 instances.

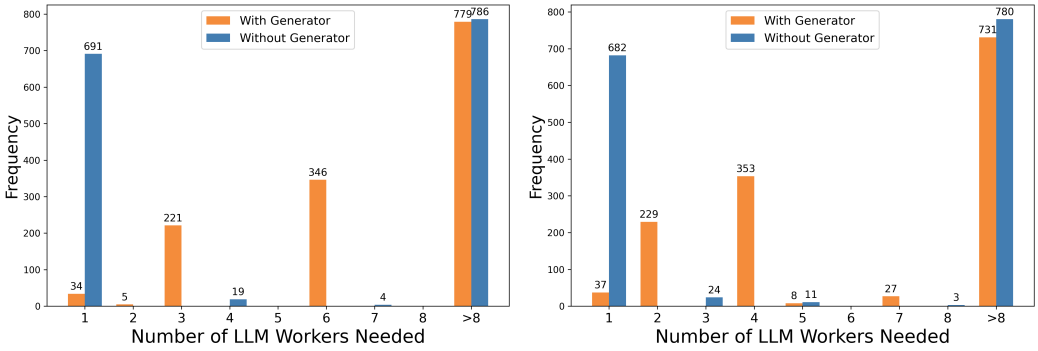
On *Offensiveness*, Figure 2.24 details LLM requirements for tolerances $\pm\{0.1, 0.2, 0.3\}$ with/without a generator trained on more than 13,000 instances. Figure 2.25 shows results for a generator trained on only 9 instances. While this lightly-trained generator saw 1337 instances needing more than 8 LLMs at ± 0.1 tolerance (more than its well-trained counterpart), it still outperformed the no-generator baseline. As tolerance loosens, many instances meet targets with 2-6 LLMs using the 9-instance generator, which requires fewer LLMs than the baseline at ± 0.3 and ± 0.4 tolerances, where the baseline still struggles.

Figures 2.26 (*QA Difficulty*) presents results with/without this 9-instance generator. On both datasets, even this lightweight generator significantly reduces the required LLMs. Without a generator, the number of instances needing >8 LLMs remains high even at relaxed tolerances (e.g., *QA Difficulty*: 750 at ± 0.3, 742 at ± 0.4). With the generator, as tolerance loosens, 1-5 LLMs resolve more instances, confirming its efficiency.



(a) Ground truth rating ±0.1

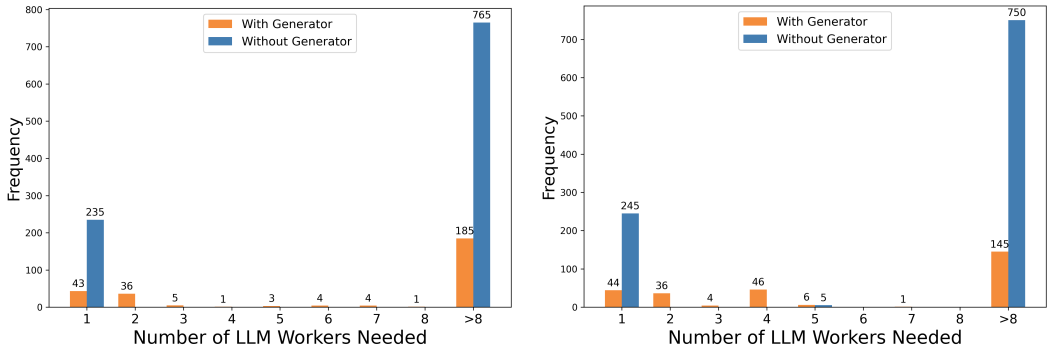
(b) Ground truth rating ±0.2



(c) Ground truth rating ±0.3

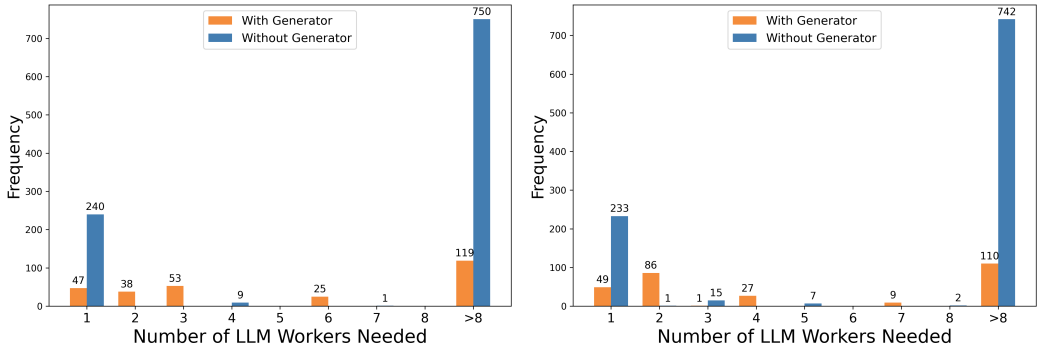
(d) Ground truth rating ±0.4

Fig. 2.25: Number of LLM workers needed to reach different performance ranges with and without generator-based rating on Offensiveness dataset. The generator is trained on only 9 instances.



(a) Ground truth rating ±0.1

(b) Ground truth rating ±0.2



(c) Ground truth rating ±0.3

(d) Ground truth rating ±0.4

Fig. 2.26: Number of LLM workers needed to reach different performance ranges with and without generator-based rating on QA Difficulty dataset.

2.7.3.6 Sensitivity Analysis

We conduct sensitivity analysis for λ in the overall loss function $\mathcal{L} = \mathcal{L}_1 + \lambda\mathcal{L}_2$, which follows the same protocol in the simulation study. We set both response error and belief diversity parameters fixed at 1, and construct a training set of 20 workers, each answering 10 randomly assigned questions. To ensure reliability, we perform 20 independent training runs for each λ value. Test performance is evaluated by aggregating predictions from 10 simulated workers per run, with confidence intervals calculated from these 20 independent runs. Fig. 2.27 demonstrate the performance results over the four evaluation metrics including MAE, RMSE, Avg. WD, and CS (same as other experiments). It is easy to see that model performance remains robust and consistently strong within the range of $[0.1, 5]$ regardless of the evaluation metric, which verifies the validity of our choice $\lambda = 1$. The results indicate our CrowdLLM is not very sensitive to the value of λ as long as it is chosen within a reasonable range. Such stability is beneficial for its practical use.

2.7.3.7 Further Exploration

The "fidelity" of the digital population may be evaluated from different dimensions. While our paper emphasizes the population-level decision-making performance instead of fine-grained individual behavioral realism, and the evaluation in our experiments is largely conducted across task instances, it is absolutely very interesting to explore how the digital populations generated by CrowdLLM perform on other different dimensions, e.g., the global statistics of the decision distributions or the individual-level patterns, compared with the human population. To this end, we conduct additional analysis on simulated data with the response error and belief diversity parameters fixed at 1. We use the same settings in the simulation study where responses from each of the 20 workers to 10 randomly assigned tasks (i.e., problems to solve) sampled from the Offensiveness dataset are used for training. The testing set contains 224 different workers, each assigned with 10 distinct tasks sampled from a task set of size 300. In our experiments, for each question, there are approximately 7.5 different workers providing answers.

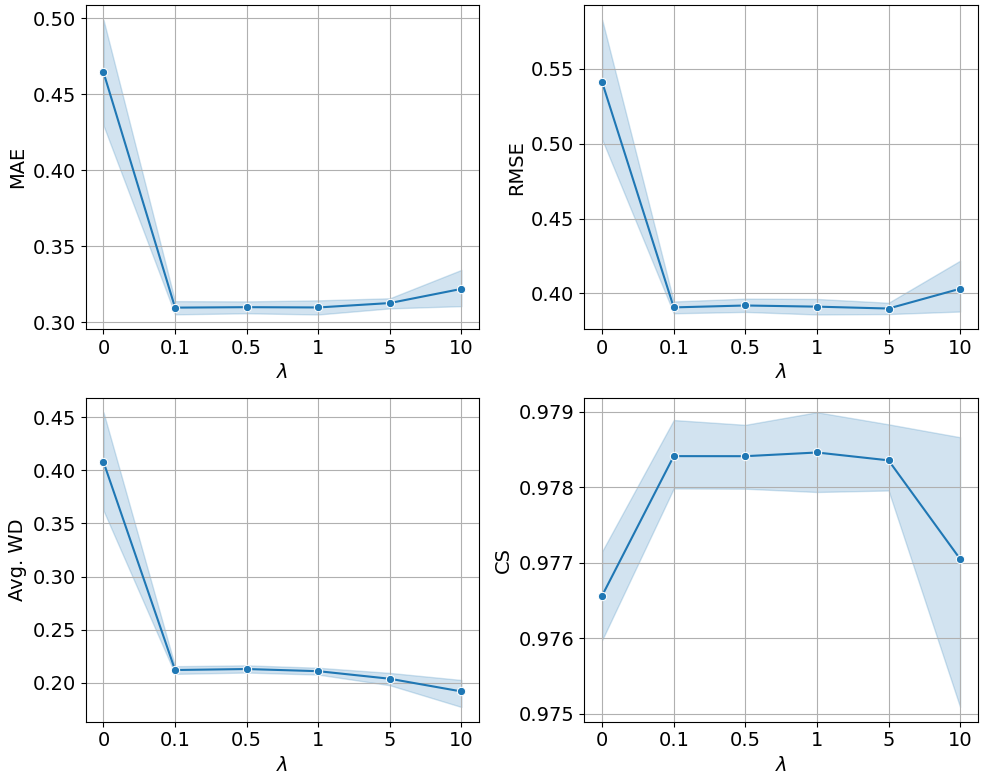


Fig. 2.27: Sensitive analysis of λ.

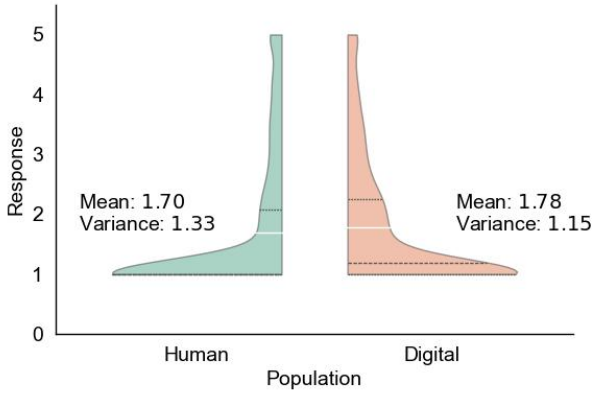
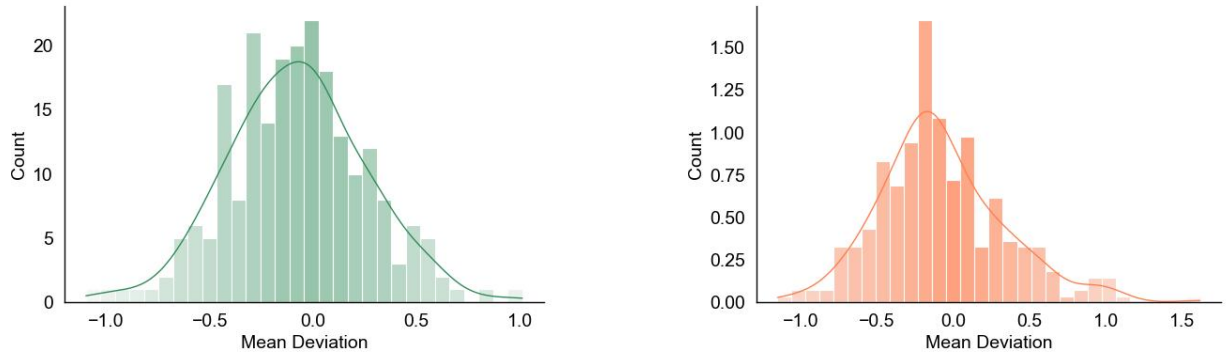


Fig. 2.28: Comparison of human and digital populations in global statistics.



(a) Mean deviation across different tasks.

(b) Mean deviation across different workers.

Fig. 2.29: Comparison of mean deviation of the digital population generated by CrowdLLM and the human population from different dimensions.

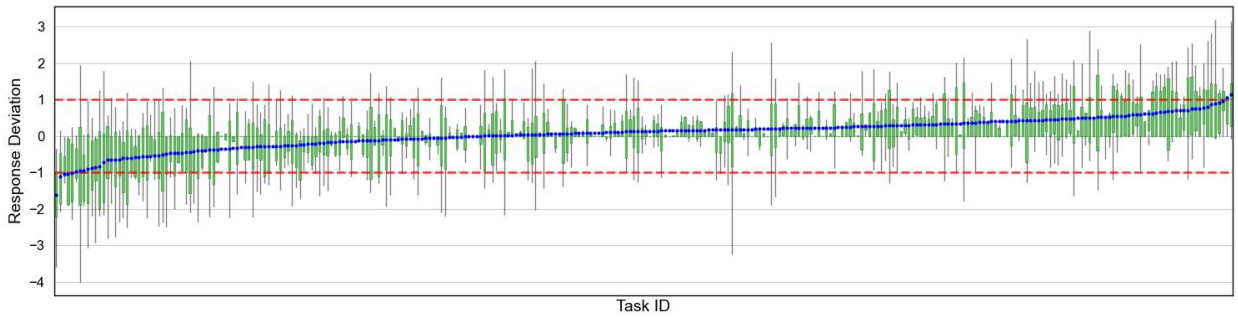


Fig. 2.30: Distributions of response deviation across different tasks.

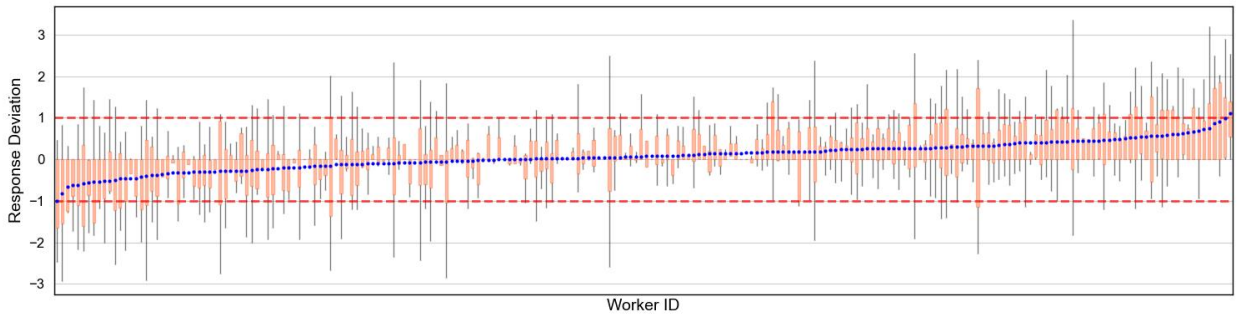


Fig. 2.31: Distributions of response deviation across different workers.

A comparison over the global first- and second-order statistics (i.e., mean and variance) between human and digital populations is demonstrated in Figure 2.28. From this comparison, the similar shapes, along with closely aligned first- and second-order statistics, indicate the agreement between the two populations. While this statistical similarity is observed at an aggregate level, it fails to reflect fine-grained decision-making patterns across different tasks or individuals. To provide a more comprehensive evaluation, Figure 2.29a and Figure 2.29b further demonstrate the gaps between the digital population and the human population in terms of mean deviation (i.e., the difference in their mean responses) across different tasks and different workers, respectively. The high concentration around zero in both figures indicates the deviation in mean responses between the digital and human populations is small, which suggests that the digital population closely matches the human population along both dimensions.

To take a closer look at the distributional fidelity, we demonstrate the full distributions of the decision deviations between the populations across different task instances (Figure 2.30) or individual workers (Figure 2.31). For both dimensions, the box plots are ordered based on the mean deviation in the responses given by the two populations (see the red dotted curves). Apparently, the mean deviations are largely centered around zero and remain within the range $[-1, 1]$. Furthermore, the majority of the boxes can be contained within this interval, which suggests that the digital population closely approximates the human population at the distributional level as well.

For a more detailed view, we further contrast the decision distributions of 20 randomly selected, matched individual pairs from the two populations, as shown in Figure 2.32. It is easy to see that, most of these workers' response medians approximate each other with absolute gaps below 1. And the decision distributions of the digital individuals look close to their human counterparts. These results provide additional evidence on the effectiveness of CrowdLLM for creating high-fidelity digital populations from a different perspective.

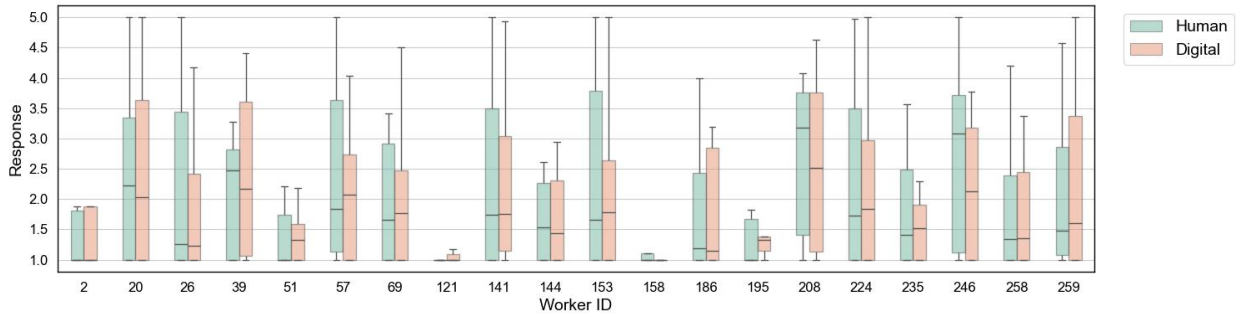


Fig. 2.32: Comparison of the decision distributions of the digital and the human populations for randomly selected individuals.

Chapter 3

FAIR COLLABORATIVE LEARNING (FAIRCL): A METHOD TO IMPROVE FAIRNESS AMID PERSONALIZATION

As we mentioned in Chapter 1 and 2, the framework of collaborative learning (CL) [85, 86, 240, 241, 86] has demonstrated remarkable efficiency in improving accuracy of individual models for various tasks. However, because of user population heterogeneity, how the individual models will perform differently for different individuals still has no answers. When users are treated distinctly to a large extent by the system, biases and unfairness will occur, which will impede its effort to build trust with users. With the increasing concern on fairness in machine learning models, it is natural to ask whether collaborative learning has any fairness guarantees. To our knowledge, there has been a lack of investigation into this question in the literature. Fairness in machine learning has emerged as a critical issue in AI ethics, due to the use of AI in high-stake decision-making applications such as law enforcement, justice, finance, education, and healthcare, etc. Fairness is broadly depicted as the absence of preference for an individual or a group with specific characteristics [62]. As we strive for model personalization, it is important to investigate whether personalization of the models could potentially exacerbate existing disparities among individuals and elevate the degree of unfairness. If so, how can we reformulate the collaborative learning method to pursue both model personalization and fairness simultaneously?

3.1 Introduction

In this chapter, we will show that model personalization does not inevitably come at the expense of fairness. Let us think of the dilemma in personalization between one-size-fits-all solution and fully individual learning (IL) (see Chapter 1). A one-size-fits-all solution only seems fair through unawareness of the different individuals. However, it does not guarantee

the same prediction accuracy for them. As a matter of fact, this one-size-fits-all approach causes substantial disparities which can result in bad model performance in practice. On the other hand, IL with no fairness considerations might also lead to such disparities. But collaborative learning was motivated to reduce this disparity [87]. In this sense, compared with the one-size-fits-all or IL approaches, CL has been already more concerned with fairness. Nonetheless, there is much room for CL to further enhance the fairness of the individual models since fairness has not been explicitly formulated in CL. In this dissertation, we propose a framework named fair collaborative learning (FairCL) that can potentially integrate a wide range of fairness concepts. We further focus on two specific fairness metrics, the bounded individual loss (BIL) and individual fairness (IF), and develop a self-adaptive algorithm for FairCL and conduct both simulated and real-world case studies. One fundamental difference between our problem and general fair machine learning is that we build many different models for personalization while the latter usually assumes one model for all groups. The remainder of the article unfolds as follows: **Section 3.2** reviews the formulation of CL and its rationale in detail. **Section 3.3** shows the CL framework is versatile to incorporate a wide range of fairness metrics. **Section 3.4** presents the self-adaptive algorithm and explore its connection with other fair CL methods. Experimental results on both simulated and real-world data will be shown in **Section 3.5** and **Section 3.6**. Conclusion is given in **Section 3.7**.

3.2 The Limitation of Collaborative Learning

In this section, we present the basic framework of collaborative learning (CL), as developed in [85, 240, 86]. This framework can be applied to both regression [87] and classification [86]. While the majority of fair machine learning works have centered on classification, in this chapter, we will also focus on classification problem to develop our models and algorithms.

3.2.1 The basic framework of CL

Suppose we have N individuals, each with their own set of observations. Let n_i denote the number of observations collected from the i-th individual. Each observation can be

represented as a pair $(\boldsymbol{x}_{ij}, y_{ij})(j = 1, \cdots, n_i)$, where $\boldsymbol{x}_{ij} \in \mathbb{R}^d$ is a feature vector and $y_{ij} \in \{0, 1\}$ is a binary label. Our task is to build a personalized model for each individual, i.e., for the i-th individual, we find a mapping $h_i : \mathbb{R}^d \to \{0, 1\}$ to predict the label y_{ij} given the feature vector $\boldsymbol{x}_{ij}$.

In an ideal scenario, each individual would have a sufficiently large data set to train an accurate personalized model for this task through fully individualized learning (IL). However, this is rarely the case in practice. While it seems more realistic to simply pool all individuals' data together to build one population-level model, this strategy could not characterize population heterogeneity, and thereby causes a great disparity on prediction accuracy among individuals. Collaborative learning (CL) offers a middle ground between these two extremes. Rather than building one population-level model, CL builds a set of canonical models that can provide an effective representation of individual models: Each individual model is characterized as a probabilistic combination of these canonical models, and this relationship is parameterized by a membership vector [86]. CL learns the parameters of the canonical models and the membership vectors of the individuals.

More specifically, let K denote the number of canonical models, where the k-th canonical model is represented as $g_k(\boldsymbol{x})$:

$$g_k(\boldsymbol{x}) = \log \frac{P(y = 1)}{P(y = 0)} = \boldsymbol{x}^\top \boldsymbol{q}_k.$$

Here, $\boldsymbol{q}_k$ denotes the unknown parameters of the k-th canonical model. For the i-th individual, their individual model can be written as

$$\log \frac{P(y_{ij} = 1|\boldsymbol{x}_{ij})}{P(y_{ij} = 0|\boldsymbol{x}_{ij})} = \sum_{k=1}^{K} c_{ik} g_k(\boldsymbol{x}_{ij}) = \boldsymbol{x}_{ij}{}^\top \sum_{k=1}^{K} c_{ik} \boldsymbol{q}_k,$$

where $c_{ik} \in [0, 1]$ is the membership parameter that characterizes how likely the model of the i-th individual comes from the k-th canonical model. The sum of $c_{ik}(k = 1, \cdots, K)$ equals 1, ensuring that the individual's model is a probabilistic combination of the canonical models. Denote $\mathbf{Q} = \begin{bmatrix} \boldsymbol{q}_1 & \cdots & \boldsymbol{q}_K \end{bmatrix}$ and $\boldsymbol{c}_i = \begin{bmatrix} c_{i1} & \cdots & c_{iK} \end{bmatrix}$. The individual model can be rewritten

as a two-parameter logistic regression model

$$p_{ij} = P(y_{ij} = 1|\boldsymbol{x}_{ij}) = \frac{\exp({\boldsymbol{x}_{ij}}^\top \mathbf{Q}\boldsymbol{c}_i)}{1 + \exp({\boldsymbol{x}_{ij}}^\top \mathbf{Q}\boldsymbol{c}_i)}.$$

The likelihood function of the CL model can be written as

$$L(\mathbf{Q}, \mathbf{C}) = \prod_{i=1}^{N}\prod_{j=1}^{n_i} p_{ij}^{y_{ij}}(1-p_{ij})^{1-y_{ij}} = \prod_{i=1}^{N}\prod_{j=1}^{n_i}\Big[\frac{\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}\boldsymbol{c}_i)}{1+\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}\boldsymbol{c}_i)}\Big]^{y_{ij}}\Big[\frac{1}{1+\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}\boldsymbol{c}_i)}\Big]^{1-y_{ij}}.$$

which gives the log-likelihood function

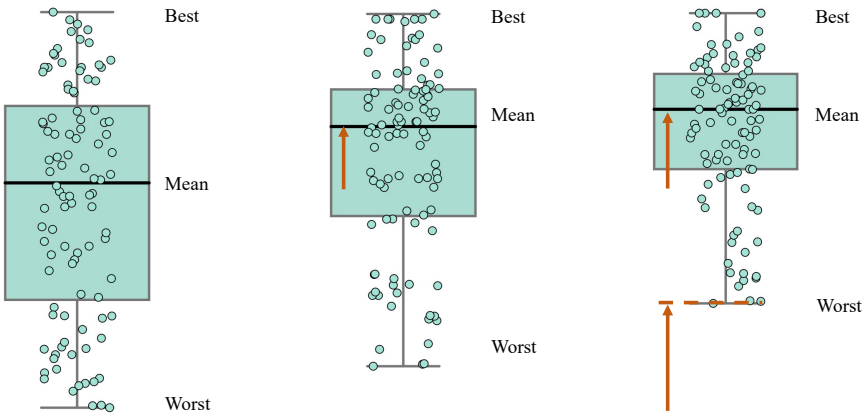
$$l(\mathbf{Q}, \mathbf{C}) = \log L(\mathbf{Q}, \mathbf{C}) = \sum_{i=1}^{N}\sum_{j=1}^{n_i}\big[y_{ij}\log\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}\boldsymbol{c}_i) - \log(1+\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}\boldsymbol{c}_i))\big].$$

The maximization of the log-likelihood is equivalent to the minimization of the negative log-likelihood loss $-l(\mathbf{Q}, \mathbf{C})$. We further normalize each individual's loss by its sample size which leads to the following optimization problem:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \frac{1}{N}\sum_{i=1}^{N}\frac{1}{n_i}\sum_{j=1}^{n_i}[\log(1+\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}\boldsymbol{c}_i)) - y_{ij}(\boldsymbol{x}_{ij}^\top \mathbf{Q}\boldsymbol{c}_i)] \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \;\; \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N. \end{aligned} \tag{3.1}$$

3.2.2 The limitation of the CL framework to achieve fairness

Compared to the fully individualized learning (IL) scheme, CL promotes fairness by enabling a high-level data sharing among the individuals and considering the canonical models as extra "data" that is fused with the individual's data, thus reducing the performance disparity among the individual models. However, CL has no explicit control over fairness. It does not incorporate any fairness constraints into the learning formulation as done in many existing fair machine learning models. Meanwhile, if we take a close look at the objective function of CL in (3.1), the goal of CL is to maximize the mean performance of the models on all the individuals (as illustrated in Figure 3.1(b)). This indicates that, to further improve fairness on CL, we can enhance the CL formulation by minimizing variance at the same time while still improving on the mean performance (as illustrated in Figure 3.1(c)). The remaining question is whether or not these two objectives would be in conflict with each other. After all, it is



(a) IL (b) CL (c) FairCL

Fig. 3.1: An illustration of the boxplots of the accuracy of individual models learned by IL (a), CL (b), and fairCL (c). CL improves on the mean level, whereas our FairCL improves on the worst cases while also improving the mean level.

commonly known in fair machine learning literature that there is a trade-off between model accuracy and fairness, and one can not increase both simultaneously. This trade-off can be quite intuitively conceived in the existing works of fair machine learning, since they build one model to serve all groups. Therefore, if one aims to increase fairness, the single model shared by all groups has to be changed, and the overall accuracy can only go down instead of going up while fairness also goes up. However, in our problem, we build personalized models rather than one single model, so this trade-off is not necessarily true. Intuitively speaking, as the current CL framework adopts a complex and nonlinear optimization problem that suffers from suboptimal solutions and overfitting, our FairCL models may help find better optimal solution and achieve better generalization, and thereby, improve both the accuracy of the models and reduce the disparity among them. Indeed, our experimental results in **Section 3.5** show that the fairness and accuracy of the personalized models can actually be improved at the same time.

Algorithm 3.1 General FairCL

Input: Training data from N users $\mathcal{D} = \{\{\mathbf{X}_i, \boldsymbol{y}_i\}_{i=1,\cdots,N}\}$.

Output: Canonical model $\mathbf{Q}$, membership $\mathbf{C}$

Initialize $\mathbf{Q}^{(0)}, \mathbf{C}^{(0)}$.
$\boldsymbol{\lambda}^{(0)} = UpdateL(\mathbf{Q}^{(0)}, \mathbf{C}^{(0)}; \mathcal{D})$.
for $t = 1, 2, \cdots, T$ **do**
    $\mathbf{Q}^{(t)} = UpdateQ(\mathbf{C}^{(t-1)}, \boldsymbol{\lambda}^{(t-1)}; \mathcal{D})$
    $\mathbf{C}^{(t)} = UpdateC(\mathbf{Q}^{(t-1)}, \boldsymbol{\lambda}^{(t-1)}; \mathcal{D})$
    if $SatisfyFairnessConstraint(\mathbf{Q}^{(t)}, \mathbf{C}^{(t)})$ **then**
        Return $\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}$
    else
        $\boldsymbol{\lambda}^{(t)} = UpdateL(\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}; \mathcal{D})$
    end if
end for
Return $\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}$

3.3 Fair Collaborative Learning (FairCL)

In this section, we explore how the CL can be integrated with existing fairness constraints to mitigate disparity among individuals and discuss the advantages and disadvantages of different fairness CL (FairCL) formulations.

3.3.1 A general framework for FairCL

The field of fairness machine learning has focused on developing general fairness constraints which are usually designed to bound a certain fairness metric on each group (i.e., the groups are defined based on some sensitive attributes such as gender and race. Then, fair learning models can be straightforwardly developed by integrating these fairness constraints into the learning formulation. In our context, it is natural to extend these fairness definitions on individuals (i.e., each individual is a "group"). A general formulation of our fair collaborative

learning (FairCL) can be written as

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} \left[\log(1+\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q} \boldsymbol{c}_i)) - y_{ij}(\boldsymbol{x}_{ij}^\top \mathbf{Q} \boldsymbol{c}_i) \right] \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \ \ \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \\ & f_i(\mathbf{C}, \mathbf{Q}) \leq 0, \quad i = 1, \cdots, N \end{aligned} \tag{3.2}$$

where $f_i(\mathbf{C}, \mathbf{Q}) \leq 0 (i = 1, \cdots, n)$ are the fairness constraints (i.e., some specific examples will be introduced in **Section 3.3** B.). It can be reformulated by introducing the Lagrangian multipliers $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_n)$ as:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} \left[\log(1+\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q} \boldsymbol{c}_i)) - y_{ij}(\boldsymbol{x}_{ij}^\top \mathbf{Q} \boldsymbol{c}_i) \right] \\ & + \sum_{i=1}^{N} \lambda_i f_i(\mathbf{C}, \mathbf{Q}) \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \ \ \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \end{aligned} \tag{3.3}$$

Algorithm 3.1 presents the general computational framework of FairCL. In the subsequent section, we will introduce various specific FairCL formulations that aim to achieve fairness by different definitions. For brevity, we will denote the empirical loss of individual i by $\ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \triangleq \frac{1}{n_i} \sum_{j=1}^{n_i} [\log(1+\exp(\boldsymbol{x}_{ij}^\top \mathbf{Q} \boldsymbol{c}_i)) - y_{ij}(\boldsymbol{x}_{ij}^\top \mathbf{Q} \boldsymbol{c}_i))]$.

3.3.2 Some specific FairCL models and algorithms

In theory, our FairCL framework shown in Eq. 3.3 is general and can incorporate a variety of fairness concept. But further modeling and computational improvement can be made while adapting this framework to a specific fairness constraint. In what follows, we present some examples of how this could be done. We will focus on widely adopted fairness metrics including:

- **Equality of Opportunity (EO)** [69]. It requires the model to assign positive outcome to the positive population with equal opportunity regardless of their groups, i.e., $P(\hat{Y} = 1|A = 0, Y = 1) = P(\hat{Y} = 1|A = 1, Y = 1)$, where A is the sensitive attribute to be

protected, Y is the label and $\hat{Y}$ is the prediction of the outcome given by the learned model.

- **Demographic Parity (DP)** [71]. It requires that the model's predictions should not depend on the sensitive attributes, i.e., $P(\hat{Y}|A=0) = P(\hat{Y}|A=1)$.

- **Fairness Through Unawareness (FTU)** [242]. It requires the sensitive attributes should not be observed or explicitly used in the model prediction.

- **Bounded Group Loss (BGL)** [73, 243]. It requires the average group loss should be bounded, i.e., $\mathbb{E}[\ell(\hat{Y}, Y)|A] \leq \zeta$, where ζ is an upper bound and $\ell(\cdot,\cdot)$ is the loss function.

- **Individual Fairness (IF)** [71]. It requires that similar individuals should be assigned with similar predictions, i.e., $D(\hat{Y}(X_1, A_1), \hat{Y}(X_2, A_2)) \leq d(X_1, X_2)$, where X_i and A_i are the insensitive and sensitive attributes of an individual, respectively. $D(\cdot)$ and $d(\cdot)$ are distance metrics for predictions $\hat{Y}$ and attributes X, respectively.

- **Counterfactual Fairness (CF)** [244]. It requires that, in both actual and counterfactual worlds where an individual has different sensitive attribute values, predictions should be the same, i.e., $P(\hat{Y}_{A\leftarrow a}(U)|X, A=a) = P(\hat{Y}_{A\leftarrow a'}(U)|X, A=a)$, where A is the sensitive attribute taking the value of a or a', U denotes the latent variables, and X denotes the insensitive attributes.

3.3.2.1 Equality of opportunity (EO) and demographic parity (DP)

While these two fairness concepts are foundational to the field of fair machine learning, their practical relevance with personalized learning is quite limited. As [245] pointed out, the DP metric might mistakenly deem a model unfair when the outcome distributions of various groups are different. This can be particularly problematic in the context of personalized

learning, where the population heterogeneity can lead to significant variations in the proportion of actual positive outcomes across individuals. In such cases, enforcing similar outcome distributions for all individuals may not necessarily improve fairness but instead result in misleading models. Only if the data distributions of individuals are very much alike (which is quite uncommon in practice), DP and EO would be practical and realistic. Therefore, in this chapter, we will not further develop algorithms or conduct experiments for these two fairness concepts but only present the FairCL formulation of DP here for theoretical interest:

$$\begin{aligned}
\min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \\
s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \;\; \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \\
& |P(h_i(\mathbf{x}; \mathbf{Q}, \mathbf{C} \geq z)|i) - P(h_i(\mathbf{x}; \mathbf{Q}, \mathbf{C}) \geq z)| \leq \xi_i, \\
& \forall z, \quad i = 1, \cdots, N.
\end{aligned}$$

Here, ξ_i is a slack variable that is a hyperparameter to create tolerance and make the formulation feasible, since in practice we will not likely achieve absolute DP.

3.3.2.2 Fairness through unawareness (FTU)

It is worthy of mentioning that the FTU actually corresponds to the one-size-fits-all approach for personalized learning.

3.3.2.3 Similarity-based individual fairness (IF)

The notion of individual fairness (IF) [71] provides a trade-off between equal treatment of individuals and individual heterogeneity. It does not necessitate absolute equality but rather mandates that individuals with similar characteristics should be treated similarly. To achieve IF, usually a similarity matrix is constructed to measure the similarity between the individuals, denoted as $\mathbf{S} = [s_{ij}]_{N \times N}$. We can obtain the Laplacian matrix $\mathbf{L}_s = \mathbf{D} - \mathbf{S}$. Here, $\mathbf{D} = \text{diag}(d_{11}, \cdots, d_{NN})$ is a $n \times n$ diagonal matrix, where $d_{ii} = \sum_{j=1}^{N} s_{ij}$. Using $\mathbf{L}_s$, it is easy to incorporate the similarity knowledge among individuals to obtain a graph-regularized

[246] formulation of CL to enforce IF:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) + \lambda Tr(\mathbf{C}^T \mathbf{L}_s \mathbf{C}) \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \;\; \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \end{aligned}$$

Actually, this formulation of FairCL-IF has been "unconsciously" developed in the prior work on collaborative learning [86], motivated not by fairness concerns but rather for accuracy improvement. However, IF alone is inadequate to guarantee protection on vulnerable individuals since it only calls for similar treatment on similar individuals [72]. Another difficulty is the construction of the similarity matrix, which usually demands a strong prior knowledge.

3.3.2.4 Bounded individual loss (BIL)

Bounded individual loss (BIL), as a natural generalization of bounded group loss (BGL) [243, 247, 248], bounds the loss function of each individual to reduce disparity. Formally, BIL is defined as

$$\mathbb{E}[\ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i)|i] \leq \zeta_i, \quad i = 1, \cdots, N.$$

The formulation of FairCL-BIL can be cast as

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \;\; \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \\ & \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \leq \zeta_i, \\ & i = 1, \cdots, N. \end{aligned} \tag{3.4}$$

To make it easier to solve, we can reformulate the problem using Lagrangian techniques. The Lagrangian function of the above formulation is

$$\ell(\mathbf{Q}, \mathbf{C}; \boldsymbol{\lambda}) = \sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) + \sum_{i=1}^{N} \lambda_i [\ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) - \zeta_i].$$

To prevent $\boldsymbol{\lambda}$ from being arbitrarily large, as demonstrated by [243], the FairCL-BIL problem can be reformulated by constraining L_1 norm of $\boldsymbol{\lambda}$ to a limit of B:

$$\begin{aligned} &\min_{\mathbf{Q},\mathbf{C}} \max_{\boldsymbol{\lambda}} \quad \ell(\mathbf{Q}, \mathbf{C}; \boldsymbol{\lambda}) \\ &s.t. \quad \boldsymbol{c}_i^\top \mathbf{1} = 1, \;\; \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \\ &\qquad ||\boldsymbol{\lambda}||_1 \leq B, \;\; \boldsymbol{\lambda} \geq 0. \end{aligned} \tag{3.5}$$

We propose to solve FairCL-BIL following the scheme in [243] that takes an exponentiated-gradient approach for optimization (Algorithm 3.2), which is an alternating iterative strategy to iterate through $\mathbf{C}$, $\mathbf{Q}$, and $\boldsymbol{\lambda}$.

3.4 Self-adaptive rearrangement by loss re-weighting

The FairCL formulations developed in **Section 3.3** show the flexibility of incorporating fairness constraints into the CL framework. Among all these possibilities, FairCL-BIL provides the most practical formulation to incorporate fairness into CL. More specifically, it provides a way to achieve fairness by controlling over the individual loss functions. When the worst individual loss is confined to a certain level, the overall loss is pushed to stay low, and the disparity between the individual losses can be shrunk by adjusting the tolerance level. Unlike DP, FairCL-BIL does not impose strict restriction, and it does not require additional information, such as the similarity matrix in IF, which can be challenging to obtain. But still, FairCL-BIL also has its limitations. The most significant one is that the formulation can be infeasible in some cases. Also, how to tune the large number of hyperparameters (such as $\boldsymbol{\zeta}$) is also a problem. To address these issues, in this section, we propose a novel approach called **R**earrangement by loss **R**e-**W**eighting (RRW) that has no feasibility issue and also enjoys easy hyperparameter tuning.

3.4.1 The basic formulation of FairCL-RRW

Instead of forcing fairness by incorporating fairness constraints into the learning formulation, here, we introduce weights $(w_1, \cdots, w_N)$ over individual loss functions and obtain a weighted

Algorithm 3.2 FairCL-BIL

Input: Training data from N users $\mathcal{D} = \{\{\mathbf{X}_i, \boldsymbol{y}_i\}_{i=1,\cdots,N}\}$, individual loss bound $\boldsymbol{\zeta}$, $\boldsymbol{\lambda}$'s upper bound B, convergence threshold ν_λ, convergence tolerance ν_ϵ.

Output: Canonical model $\mathbf{Q}$, membership $\mathbf{C}$

Initialize $\mathbf{Q}$, $\mathbf{C}$.
Set $\boldsymbol{\theta}^{(\mathbf{1})} = \mathbf{0} \in \mathbb{R}^N$.
for $t = 1, 2, \cdots, T$ **do**
▷ Initialize $\boldsymbol{\lambda}^{(t)}$
    for $i = 1, \cdots, N$ **do**
        $\lambda_i^{(t)} = \frac{B \exp(\theta_i^{(t)})}{1+\sum_{j=1}^N \exp(\theta_j^{(t)})}$
    end for
▷ Update $\mathbf{Q}^{(t)}$
    $\mathbf{Q}^{(t)} = \arg\min_{\mathbf{Q}} \ell(\mathbf{C}, \mathbf{Q}; \boldsymbol{\lambda}^{(t)})$
▷ Update $\mathbf{C}^{(t)}$
    $\mathbf{C}^{(t)} = \arg\min_{\boldsymbol{c}_i^\top \mathbf{1}=1, \boldsymbol{c}_i \geq 0} \ell(\mathbf{C}, \mathbf{Q}; \boldsymbol{\lambda}^{(t)})$
▷ Update $\boldsymbol{\lambda}^{(t)}$
    $\boldsymbol{\lambda}_*^{(t)} = \arg\max_{||\boldsymbol{\lambda}||_1 \leq B, \boldsymbol{\lambda} \geq 0} \ell(\mathbf{C}^{(t)}, \mathbf{Q}^{(t)}; \boldsymbol{\lambda})$
    $\overline{\nu}_\lambda = \ell(\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}; \boldsymbol{\lambda}_*^{(t)}) - \ell(\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}; \boldsymbol{\lambda}^{(t)})$
    $\overline{\nu}_\epsilon = \ell(\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}; \boldsymbol{\lambda}) - \ell(\mathbf{Q}^{(t-1)}, \mathbf{C}^{(t-1)}; \boldsymbol{\lambda})$
    if $\overline{\nu}_\lambda \leq \nu_\lambda$ **and** $\overline{\nu}_\epsilon \leq \nu_\epsilon$ **then**
        if $PassFairConstraint(\mathbf{Q}^{(t)}, \mathbf{C}^{(t)})$ **then**
            Return $\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}$
        else
            Return **Ø**
        end if
    end if
    $\boldsymbol{l}^{(\boldsymbol{t})} = (\ell_1(\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}), \cdots, \ell_N(\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}))$
    Set $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \eta(\boldsymbol{l}^{(\boldsymbol{t})} - \boldsymbol{\zeta})$
end for

optimization problem, which leads to the formulation of FairCL-RRW:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} w_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \;\; \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N. \end{aligned} \tag{3.6}$$

Note that, here, the weights $\boldsymbol{w}$ are not fixed but rather iteratively updated (i.e., as shown in Algorithm 3.4) in order to dynamically rebalance the weights of the individual losses. This allows the individual models who have larger losses in the last iteration to be further minimized in the next iteration. The essence of RRW does not lie in the weighted form of the objective function, but in the automatic updates of the weights during the optimization process.

3.4.2 Automatic re-weighting by a self-adaptive strategy

To make sure the iterative update of $\boldsymbol{w}$ can rebalance the individual losses and drive the optimization process more towards on the improvement on the "worse cases" as shown in Figure 3.1(c), the following **Lemma 1** provides a good hint about how this rebalancing process can be systematically and automatically managed by a simple self-adaptive strategy of re-weighting.

Lemma 3.4.1 (Rearrangement Inequality)**.** *Given the ascending sequences of weights* $w_1 \leq \cdots \leq w_N$ *and values* $f_1 \leq \cdots \leq f_N$*, we have*

$$\sum_{i=1}^{N} w_i f_{N-i+1} \leq \sum_{i=1}^{N} w_i f_{\sigma(i)} \leq \sum_{i=1}^{N} w_i f_i$$

where $\sigma(1), \cdots, \sigma(N)$ *is a random permutation of* $1 \sim N$*.*

As our optimization problem is a minimization problem, this lemma suggests that if we set up the weights in the same ordering as the individual losses, the minimization problem in Eq. (3.6) will force a re-ordering of the individual losses, in a reverse order of the weights. In other words, it will force the larger losses to be smaller in the next iteration.

Motivated by this idea, our proposed strategy is that, in each iteration t, the assigned weight w_i for each individual is an increasing function of the contribution of the individual

loss to the total loss in the last iteration, i.e., $w_i^t = g(\frac{\ell_i^{(t-1)}}{\sum_{j=1}^N \ell_j^{(t-1)}}) > 0$. And one of the simplest re-weighting functions $g(\cdot)$ is the power function given in Algorithm 3.4, i.e., $g(x) = x^q$. The following theorems show the rationale of using Algorithm 3.4 for re-weighting:

Theorem 3.4.2. *FairCL-RRW reweighted by Algorithm 3.4 is equivalent to solving a min-max problem with the objective function* $\min_{\mathbf{C},\mathbf{Q}} \max_{\boldsymbol{w}} \sum_{i=1}^N w_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i)$.

Proof. To show the equivalence, we consider the following min-max problem:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \max_{\boldsymbol{w}} \quad & \sum_{i=1}^N w_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \quad \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \\ & \sum_{i=1}^N w_i^p = 1, w_i > 0, \forall i = 1, \cdots, N. \end{aligned} \tag{3.7}$$

Here, $p > 1$ and the L_p norm of the weights is bounded. Then, for the inner-level maximization, the equivalent Lagrangian formulation can be written as

$$\begin{aligned} \max_{\boldsymbol{w}} \quad & \sum_{i=1}^N w_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) - \lambda(\sum_{i=1}^p w_i^p - 1) \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \quad \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \\ & . \end{aligned} \tag{3.8}$$

where the objective function is $f(\boldsymbol{w}) = \sum_{i=1}^N w_i \ell_i - \lambda(\sum_{i=1}^p w_i^p - 1)$. Noticing the convexity of $f(\cdot)$, the maximal point should be reached when $\frac{\partial f}{\partial \boldsymbol{w}} = 0$. It then leads to

$$0 = \frac{\partial f}{\partial w_i} = \ell_i - \lambda p w_i^{p-1}, \quad \forall i = 1, \cdots, N.$$

which indicates $w_i \propto \ell_i^{\frac{1}{p-1}}$. Since $p > 1$, it is easy to see that $\frac{1}{p-1} > 0$. It should be mentioned that we do not really care about the scaling factor in the optimization due to the symmetry that all the weights will be affected equally by this factor. Thus, we can always find $q = \frac{1}{p-1}$ and update through $w_i = \frac{\ell_i^q}{g(\ell_1, \cdots, \ell_N)}$ to normalize the weights so as to satisfy the constraints. However, the magnitude of the weights will not affect the minimization process,

which indicates flexibility in the choice of the normalization factor. So it is better to update $\boldsymbol{w}$ in a way that makes it easier to solve the problem. Obviously, $g(\ell_1, \cdots, \ell_N) = (\ell_1 + \cdots + \ell_N)^q$ is symmetric and also it has some computational simplicity in practice. So we can use it for the normalization, which results in the reweighting function in Algorithm 3.4. Therefore, the alternating optimization of the min-max problem (3.7) is equivalent to Algorithm 3.3 reweighted by the power function in Algorithm 3.4. □

Theorem 3.4.2 reveals the nature of FairCL-RRW as demonstrated in Figure 3.1 (c) that aims to improve on the worse cases. I.e., larger weight will be assigned to the individual with worse performance so that its impact on the overall performance will be amplified in the next iteration. Note that a key element in the automatic re-weighting method in FairCL-RRW is the power parameter $q(> 0)$, which regulates the balance between individual heterogeneity and commonality. The following theorem further uncovers some nature of this parameter:

Theorem 3.4.3. *When $q = 1$, the update of $\mathbf{Q}$ and $\mathbf{C}$ in Algorithm 3.3 aims to shrink the variance of the individual losses at the cost of the loss change incurred in the update.*

Proof. As mentioned in the proof of Theorem 3.4.2, the scaling factor for $\boldsymbol{w}$ does not actually affect optimization. Thus, given that $\sum_{i=1}^N \ell_i$ is fixed in each reweighting iteration for the update of $\mathbf{Q}$ and $\mathbf{C}$, we can ignore it and equivalently consider $w_i = \ell_i^q$. Suppose in iteration t, after updating $\mathbf{Q}$ and $\mathbf{C}$, the individual i's loss is represented by $\ell_i^{(t)}$. When $q = 1$, the updating procedure can be formulated as

$$\ell_1^{(t+1)}, \ldots, \ell_N^{(t+1)} = \underset{\ell_1, \cdots, \ell_N}{\arg\max} \sum_{i=1}^N \ell_i^{(t)} \ell_i.$$

We notice that

$$\begin{aligned}\sum_{i=1}^N \ell_i^{(t)} \ell_i^{(t+1)} &= \sum_{i=1}^N \ell_i^{(t)} \ell_i^{(t+1)} - \frac{1}{N} \sum_{i=1}^N \ell_i^{(t)} \sum_{j=1}^N \ell_j^{(t+1)} + \frac{1}{N} \sum_{i=1}^N \ell_i^{(t)} \sum_{j=1}^N \ell_j^{(t+1)} \\ &= \sum_{i=1}^N (\ell_i^{(t)} - \frac{1}{N} \sum_{j=1}^N \ell_j^{(t)})(\ell_i^{(t+1)} - \frac{1}{N} \sum_{j=1}^N \ell_j^{(t+1)}) + \frac{1}{N} \sum_{i=1}^N \ell_i^{(t)} \sum_{j=1}^N \ell_j^{(t+1)}\end{aligned}$$

$$
\begin{aligned}
=&\frac{1}{2}\{\sum_{i=1}^{N}(\ell_i^{(t)}-\frac{1}{N}\sum_{j=1}^{N}\ell_j^{(t)})^2+\sum_{i=1}^{N}(\ell_i^{(t+1)}-\frac{1}{N}\sum_{j=1}^{N}\ell_j^{(t+1)})^2-\sum_{i=1}^{N}[(\ell_i^{(t)}-\frac{1}{N}\sum_{j=1}^{N}\ell_j^{(t)})\\
&-(\ell_i^{(t+1)}-\frac{1}{N}\sum_{j=1}^{N}\ell_j^{(t+1)})]^2\}+\frac{1}{N}\sum_{i=1}^{N}\ell_i^{(t)}\sum_{j=1}^{N}\ell_j^{(t+1)}\\
=&\frac{1}{2}\{\mathrm{Var}^{(t)}+\mathrm{Var}^{(t+1)}-\sum_{i=1}^{N}[(\ell_i^{(t)}-\ell_i^{(t+1)})^2+(\frac{1}{N}\sum_{j=1}^{N}\ell_j^{(t)}-\frac{1}{N}\sum_{j=1}^{N}\ell_j^{(t+1)})^2\\
&-2(\ell_i^{(t)}-\ell_i^{(t+1)})(\frac{1}{N}\sum_{j=1}^{N}\ell_j^{(t)}-\frac{1}{N}\sum_{j=1}^{N}\ell_j^{(t+1)})]\}+\frac{1}{N}\sum_{i=1}^{N}\ell_i^{(t)}\sum_{j=1}^{N}\ell_j^{(t+1)}\\
=&\frac{1}{2}[\mathrm{Var}^{(t)}+\mathrm{Var}^{(t+1)}+\frac{1}{N}(\sum_{i=1}^{N}\ell_i^{(t)})^2+\frac{1}{N}(\sum_{i=1}^{N}\ell_i^{(t+1)})^2-\sum_{i=1}^{N}(\ell_i^{(t)}-\ell_i^{(t+1)})^2].
\end{aligned}
$$

□

Given that when updating $\mathbf{Q}$ and $\mathbf{C}$ in the iteration t, i.e., updating $\ell_1^{(t+1)},\cdots\ell_N^{(t+1)}$, the weights $\ell_1^{(t)},\cdots\ell_N^{(t)}$ are fixed, it is obvious that

$$
\min_{\mathbf{Q},\mathbf{C}}\sum_{i=1}^{N}\ell_i^{(t)}\ell_i^{(t+1)}(\mathbf{X}_i,\boldsymbol{y}_i;\mathbf{Q},\boldsymbol{c}_i)\iff\min\mathrm{Var}^{(t+1)}+\frac{1}{N}(\sum_{i=1}^{N}\ell_i^{(t+1)})^2-||\boldsymbol{\ell}^{(t)}-\boldsymbol{\ell}^{(t+1)}||_2^2.
$$

For the objective function, there are three parts: the variance of the individual losses (i.e., $\mathrm{Var}^{(t+1)}$), the square of the basic CL loss (i.e., $(\sum_{i=1}^{N}\ell_i^{(t+1)})^2$), and the distance between the losses in the last iteration and the losses in this iteration. Thus, Algorithm 3.3 aims to simultaneously minimize the variance of the individual losses and the basic CL loss, however, at the cost of the loss change.

It is worthy of mentioning that, when $q\neq 1$, Algorithm 3.3 still shrinks the "variation" of the individual losses. It is just when $q=1$, the shrunk variation coincides with the exact form of variance (i.e., as a quadratic form). Actually, when q is large, the algorithm will pay more attention to the inter-individual differences and focus on improving the low-achieving models to achieve better equity. In contrast, when q is small, it tends to create equal weights for all individuals and treat their loss functions indifferently in optimization. In practice, we tune q through cross validation with a grid search.

3.4.3 Enhance the stability of FairCL-RRW by temporal smoothing (TS) regularization

In the proof of **Theorem 3.4.3**, a byproduct is that we find out that the objective function which Algorithm 3.3 aims to minimize when updating $\mathbf{Q}$ and $\mathbf{C}$ can be equivalently decomposed into three parts:

$$\text{Var}^{(t+1)} + \frac{1}{N}(\sum_{i=1}^{N} \ell_i^{(t+1)})^2 - ||\boldsymbol{\ell}^{(t)} - \boldsymbol{\ell}^{(t+1)}||_2^2,$$

where the first term is the variance of the individual losses and the second term is proportional to the square of the basic CL loss $\sum_{i=1}^{N} \ell_i^{(t+1)}$, which makes sense. But the third term is to maximize the difference between the losses in current iteration and the losses in the last iteration $||\boldsymbol{\ell}^{(t)} - \boldsymbol{\ell}^{(t+1)}||_2$. Obviously, it is undesired that $||\boldsymbol{\ell}^{(t)} - \boldsymbol{\ell}^{(t+1)}||_2$ gets larger since it might lead to the instability of the optimal solution if the solution drastically changes from iteration to iteration. To eliminate this potential instability, a natural idea is to consider an additional term to offset it. It is not easy to directly penalize this distance term in consideration of the complexity of $\boldsymbol{l}$. Instead, we propose a solution that is based on the following theorem:

Theorem 3.4.4. *The loss $\boldsymbol{\ell}^{(t)}$ is temporally Lipschitz continuous on $\mathbf{Q}$ and $\mathbf{C}$.*

Proof. Due to the Lipschitz continuity of the logarithm function and the Sigmoid function, there must exist M_{ij} and $L_{ij} (i = 1, \cdots, N, \forall i, j = 1, \cdots, n_i)$ such that

$$\begin{aligned}
&||\boldsymbol{\ell}^{(t)} - \boldsymbol{\ell}^{(t+1)}||_2^2 \\
=&\sum_{i=1}^{N} \left|\ell_i^{(t)}(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) - \ell_i^{(t+1)}(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i)\right|^2 \\
=&\sum_{i=1}^{N} \frac{1}{n_i} \Big| \sum_{j=1}^{n_i} \log(\frac{\exp(y_{ij} \boldsymbol{x}_{ij}^\top \mathbf{Q}^{(t)} \boldsymbol{c}_i^{(t)})}{1 + \exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}^{(t)} \boldsymbol{c}_i^{(t)})}) - \sum_{j=1}^{n_i} \log(\frac{\exp(y_{ij} \boldsymbol{x}_{ij}^\top \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t+1)})}{1 + \exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t+1)})}) \Big|^2 \\
\leq&\sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} \Big| \frac{\exp(y_{ij} \boldsymbol{x}_{ij}^\top \mathbf{Q}^{(t)} \boldsymbol{c}_i^{(t)})}{1 + \exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}^{(t)} \boldsymbol{c}_i^{(t)})} - \frac{\exp(y_{ij} \boldsymbol{x}_{ij}^\top \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t+1)})}{1 + \exp(\boldsymbol{x}_{ij}^\top \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t+1)})} \Big|^2
\end{aligned}$$

$$
\begin{aligned}
&\leq \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} ||\mathbf{Q}^{(t)} \boldsymbol{c}_i^{(t)} - \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t+1)}||_2^2 \\
&= \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} ||\mathbf{Q}^{(t)} \boldsymbol{c}_i^{(t)} - \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t)} + \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t)} - \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t+1)}||_2^2 \\
&\leq \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} (||\mathbf{Q}^{(t)} \boldsymbol{c}_i^{(t)} - \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t)}||_2^2 + ||\mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t)} - \mathbf{Q}^{(t+1)} \boldsymbol{c}_i^{(t+1)}||_2^2) \\
&\leq \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} (||\mathbf{Q}^{(t)} - \mathbf{Q}^{(t+1)}||_F^2 ||\boldsymbol{c}_i^{(t)}||_2^2 + ||\mathbf{Q}^{(t+1)}||_F^2 ||\boldsymbol{c}_i^{(t)} - \boldsymbol{c}_i^{(t+1)}||_2^2) \\
&\leq \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} (K^2 ||\mathbf{Q}^{(t)} - \mathbf{Q}^{(t+1)}||_F^2 + K(d+1) T_Q ||\boldsymbol{c}_i^{(t)} - \boldsymbol{c}_i^{(t+1)}||_2^2) \\
&= ||\mathbf{Q}^{(t)} - \mathbf{Q}^{(t+1)}||_F^2 \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} K^2 + \sum_{i=1}^{N} ||\boldsymbol{c}_i^{(t)} - \boldsymbol{c}_i^{(t+1)}||_2^2 \frac{K(d+1) T_Q}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} \\
&\leq ||\mathbf{Q}^{(t)} - \mathbf{Q}^{(t+1)}||_F^2 \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} K^2 + \sum_{i=1}^{N} ||\boldsymbol{c}_i^{(t)} - \boldsymbol{c}_i^{(t+1)}||_2^2 \max_i \Big\{ \frac{K(d+1) T_Q}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} \Big\} \\
&= ||\mathbf{Q}^{(t)} - \mathbf{Q}^{(t+1)}||_F^2 \sum_{i=1}^{N} \frac{1}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} K^2 + ||\mathbf{C}^{(t)} - \mathbf{C}^{(t+1)}||_F^2 \max_i \Big\{ \frac{K(d+1) T_Q}{n_i} \sum_{j=1}^{n_i} M_{ij} L_{ij} \Big\} \\
&= \lambda_1 ||\mathbf{Q}^{(t)} - \mathbf{Q}^{(t+1)}||_F^2 + \lambda_2 ||\mathbf{C}^{(t)} - \mathbf{C}^{(t+1)}||_F^2
\end{aligned}
$$

where T_Q is the upper bound of the elements in $\mathbf{Q}$, $||\cdot||_2$ is L_2 norm and $||\cdot||_F$ is Frobenius norm. Here, the first inequality (the third line) and the second inequality are obtained by the Lipschitz continuity of logarithm function and Sigmoid function, respectively. The third inequality (the sixth line) results from the triangle inequality, and the fourth inequality (the seventh line) results from $||\mathbf{A}\boldsymbol{x}|| \leq ||A||_F ||\boldsymbol{x}||_2$. The fifth inequality is obtained based on our assumption that elements in $\mathbf{Q}$ are bounded and $\mathbf{C}$ satisfies the probability constraint in the formulation. As a result, we see $||\boldsymbol{\ell}^{(t)} - \boldsymbol{\ell}^{(t+1)}||_2^2$ is bounded by the linear combination of $||\mathbf{Q}^{(t)} - \mathbf{Q}^{(t+1)}||_F^2$ and $||\mathbf{C}^{(t)} - \mathbf{C}^{(t+1)}||_F^2$. This indicates the Lipschitz continuity, which concludes the proof. □

Theorem 3.4.4 gives the upper bound of $||\boldsymbol{\ell}^{(t)} - \boldsymbol{\ell}^{(t+1)}||$. Thus, instead of incorporating

Algorithm 3.3 FairCL-RRW

Input: Training data from $\{\{\mathbf{X}_i, \boldsymbol{y}_i\}_{i=1,\cdots,N}\}$.

Output: Canonical model $\mathbf{Q}$, membership $\mathbf{C}$

1: Initialize $\mathbf{Q}^{(0)}, \mathbf{C}^{(0)}$.
2: **for** $t = 1, 2, \cdots, T$ **do**
3: **for** $i = 1, 2, \cdots, N$ **do**
4: Compute the individual loss $\ell_i^{(t-1)}$ for individual i given $\mathbf{Q}^{(t-1)}$, $\mathbf{C}^{(t-1)}$.
5: **end for**
6: Update $\boldsymbol{w}^{(t)} = \text{Reweight}(\ell_1^{(t-1)}, \cdots, \ell_N^{(t-1)})$.
7: Update $\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}$ by minimizing (3.9) given $\boldsymbol{w}^{(t)}$.
8: **end for**
9: Return $\mathbf{Q}^{(t)}, \mathbf{C}^{(t)}$

$||\boldsymbol{\ell}^{(t)} - \boldsymbol{\ell}^{(t+1)}||$ directly into formulation (3.6), we propose a temporal smooth (TS) regularization on the change of $\mathbf{Q}$ and $\mathbf{C}$ over time. With this temporal smoothing regularization, we can revise the formulation (3.6) with the following formulation:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} w_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) + \lambda_1 ||\mathbf{Q}^{(t)} - \mathbf{Q}^{(t+1)}||_F^2 \\ & + \lambda_2 ||\mathbf{C}^{(t)} - \mathbf{C}^{(t+1)}||_F^2 \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \;\; \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \end{aligned} \tag{3.9}$$

The regularization parameters λ_1 and λ_2 can be tuned by cross validation. In **Section 3.5**, we will show that this TS regularization strategy can lead to a great stability of the FairCL-RRW formulation.

3.4.4 Connection with the FairCL-BIL

The following theorem builds connection between FairCL-BIL and FairCL-RRW.

Theorem 3.4.5. *There exist $w_1, \ldots, w_n$ such that the optimal solution to* (3.4) *is also optimal to* (3.6).

Algorithm 3.4 Reweight

Input: Individual losses $\ell_1, \ell_2 \cdots, \ell_N$, hyperparameter $q > 0$.

Output: Individual weights $\boldsymbol{w}$

1: **for** $i = 1, 2, \cdots, N$ **do**

2: $\quad w_i = (\frac{\ell_i}{\sum_{j=1}^{N} \ell_j})^q$

3: **end for**

4: $\boldsymbol{w} = [w_1, w_2, \cdots, w_N]$

5: Return $\boldsymbol{w}$

Proof. For $i = 1, \cdots, n$, let $a_i = \frac{z_i}{\ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^*, \mathbf{C}^*)}$, where $z_i \geq \zeta_i$. Consider the following problem:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} a_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \\ s.t. \quad & \boldsymbol{c}_i^\top \mathbf{1} = 1, \;\; \boldsymbol{c}_i \geq 0, \quad i = 1, \ldots, N \\ & a_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \leq \zeta_i, \quad i = 1, \ldots, N \end{aligned} \tag{3.10}$$

Suppose that the optimal solution to (3.10) is $\mathbf{Q}^{'}, \mathbf{C}^{'}$. To satisfy the bound constraints, we must have

$$\zeta_i \geq a_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^{'}, \boldsymbol{c}_i^{'}) = \frac{z_i}{\ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^*, \boldsymbol{c}_i^*)} \cdot \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^{'}, \boldsymbol{c}_i^{'}) \geq \frac{\zeta_i}{\ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^*, \boldsymbol{c}_i^*)} \cdot \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^{'}, \boldsymbol{c}_i^{'})$$

which leads to

$$\ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^{'}, \boldsymbol{c}_i^{'}) \leq \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^*, \boldsymbol{c}_i^*)$$

Thus, we have

$$\sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^{'}, \boldsymbol{c}_i^{'}) \leq \sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^*, \boldsymbol{c}_i^*) \tag{3.11}$$

Given that $\mathbf{Q}^*, \mathbf{C}^*$ is the optimal solution to (3.4), we should have

$$\sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^*, \boldsymbol{c}_i^*) \leq \sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^{'}, \boldsymbol{c}_i^{'}) \tag{3.12}$$

Combining (3.11) and (3.12), a resulting equality can be built as

$$\sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^{'}, \boldsymbol{c}_i^{'}) = \sum_{i=1}^{N} \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}^{*}, \boldsymbol{c}_i^{*})$$

This indicates $\mathbf{Q}^{'}, \mathbf{C}_i^{'}$ is also an optimal solution to (3.4), which builds the equivalence of the optimal solution between (3.10) and (3.4). On the other hand, given that the linear transformation of the objective function will not affect the optimal solution, we can introduce a large number M such that (3.10) can be equivalently written as:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & M \sum_{i=1}^{N} a_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \\ s.t. \quad & \boldsymbol{c}_i^{\top} \mathbf{1} = 1, \quad \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \\ & a_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \leq \zeta_i, \quad i = 1, \cdots, N \end{aligned} \tag{3.13}$$

Let $w_i = Ma_i$. Then (3.13) can be rewritten as

$$\begin{aligned} \min_{\mathbf{C},\mathbf{Q}} \quad & \sum_{i=1}^{N} w_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \\ s.t. \quad & \boldsymbol{c}_i^{\top} \mathbf{1} = 1, \quad \boldsymbol{c}_i \geq 0, \quad i = 1, \cdots, N \\ & w_i \ell_i(\mathbf{X}_i, \boldsymbol{y}_i; \mathbf{Q}, \boldsymbol{c}_i) \leq M\zeta_i, \quad i = 1, \cdots, N \end{aligned}$$

Since M can be arbitrarily large, we can remove the bound constraint, which results in (3.6), i.e., the optimal solution to (3.10), $\mathbf{Q}^{'}, \mathbf{C}^{'}$, is also the optimal solution to (3.6). Thus, we can conclude that the optimal solution to (3.4) is also the optimal solution to (3.6). □

This theorem shows that, in theory, the best performance of FairCL-BIL can be achieved by FairCL-RRW if the weights are appropriately set. Meanwhile, when FairCL-BIL is infeasible, FairCL-RRW can still be feasible. Therefore, we can expect that with fine-tuned weights, FairCL-RRW could in theory always outperform FairCL-BIL, which will be verified by our numerical results in **Section 3.5**.

3.4.5 Computational efficiency

To learn personalized models of N individuals, given the number of features d and the number of canonical models K ($K < d, N$, usually set small values), while IL will require $N \cdot d$ parameters, FairCL methods can reduce the number of parameters to $K \cdot d + N \cdot K$. It can significantly save a space with $O(K(N + d) + N)$ compared to IL when K is far smaller than d and N. However, in each iteration, FairCL-RRW needs to deal with more parameters than the separate training in IL. Assume the sample size for each individual is m. The update of $\mathbf{Q}$ requires a cost of $O(NKmd)$ while the update of $\mathbf{C}$ needs $O(NKmd + NKm)$. Given the cost of re-weighting time is linear to N individuals, the total time complexity can be roughly written as $O(TKNmd)$, where T is the number of iterations. Apparently, running FairCL-RRW may incur more computational cost than IL. But given that K is usually small, the cost of each iteration should be comparable to IL. And in practice, the algorithm can usually converge to a good solution in a few iterations. It should be mentioned that, due to the matrix operations, when the data is large, there might be some numerical issues. Currently, our methods can efficiently deal with data from hundreds or of individuals, but when the dataset or resulting optimization problem is too large, we might need more efficient and scalable computation strategies, which is one of our future direction.

In the following sections, we conduct numerical evaluations of the proposed FairCL formulations using both simulated and real-world data. The real-world data includes datasets collected from three different applications. We assess the performance of the algorithms in terms of both prediction accuracy and fairness, i.e., the disparity between the prediction accuracy of the individual models evaluated by variance, and the accuracy of low-achieving individual models. All algorithms are implemented in Python3 on MacBook Pro with Apple M1 chips.

3.5 Simulation studies

We simulate data from $N = 100$ individuals. For individual i, we randomly generate a sample size n_i by adding 5 to n_i' which is uniformly drawn from $\{1, 2, \cdots, 50\}$. Here, we set

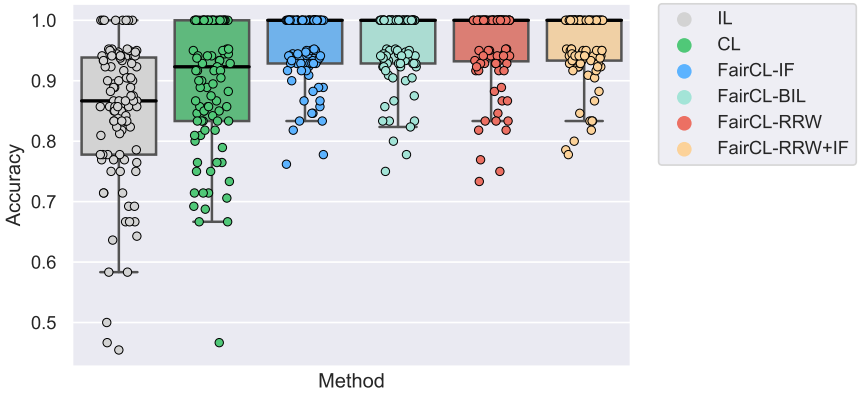
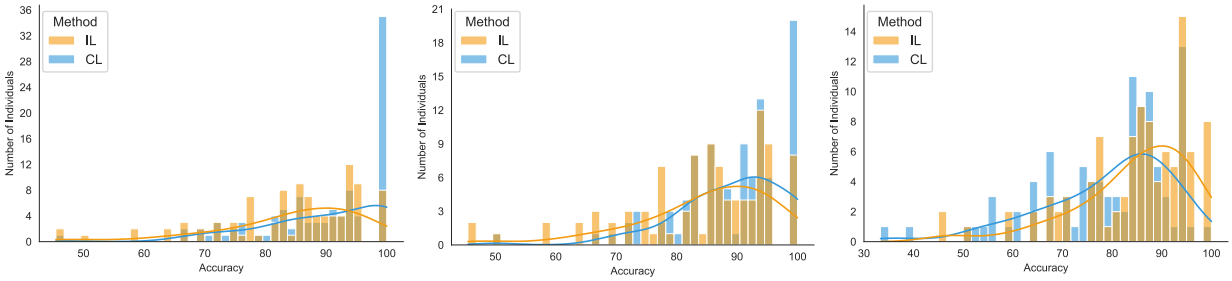


Fig. 3.2: Comparison of different methods (d=10).

$n_i > 5$ to ensure that each individual has a minimum number of samples to enable model training. Following the setup in [86], we generate the canonical models encoded in $\mathbf{Q}$ and the individual membership mappings encoded in $\mathbf{C}$ separately. In our data generation, we fix the true number of canonical models to $K = 3$. And we choose the dimension of the data d from $\{5, 10, 100\}$. Given K and d, $\mathbf{Q} = [q_{ij}]_{(d+1)\times K}$ is randomly generated from the standard normal distribution, i.e., $q_{ij} \sim \mathcal{N}(0, 1)$. To generate $\mathbf{C}$, we consider two generation strategies.

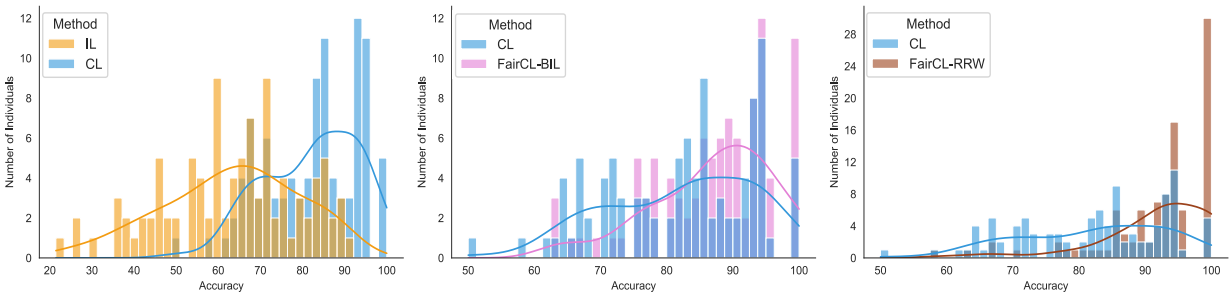
- **Totally random strategy.** With this strategy, for each individual i, we draw $\boldsymbol{c}_i$ from a normal distribution, i.e., $\boldsymbol{c}_i \sim N(\mathbf{0}, \mathbf{I}_{K\times K})$. Then, they are normalized so that $\boldsymbol{c}_i^\top \mathbf{1} = 1$ and concatenated into a matrix $\mathbf{C} = [\boldsymbol{c}_1, \cdots, \boldsymbol{c}_N]$.

- **Hierarchical strategy.** For each individual, first, k_i is randomly drawn from $\{1, \cdots, K\}$ by the uniform distribution, which represents that the k_ith model is chosen. Then, the individual membership $\boldsymbol{c}_i$ is drawn following $\boldsymbol{c}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{E}_{k_i})$, where the variance $\mathbf{E}_k = \text{diag}(1, \cdots, 1, \sigma^2, 1, \cdots, 1)$ is an almost identity matrix with the kth entry replaced by σ^2. Just like the totally random strategy, we also normalize the vectors before concatenation.



(a) $\hat{K}$=3 (b) $\hat{K}$=5 (c) $\hat{K}$=10

Fig. 3.3: Comparison of IL and CL on simulated data under a hierarchical strategy (d=10).



(a) IL v.s. CL (b) CL v.s. FairCL-BIL (c) CL v.s. FairCL-RRW

Fig. 3.4: Comparison of test accuracy distributions on simulated data under a hierarchical strategy without prior knowledge ($\hat{K} = 3, d = 100$).

With $\mathbf{Q}$ and $\mathbf{C}$ being generated, the individual models can be easily obtained by $\boldsymbol{\beta}_i = \mathbf{Q}\boldsymbol{c}_i$. We then generate their datasets $\mathcal{D}_i = \{(\boldsymbol{x}_{ij}, y_{ij})\}_{j=1,\cdots,n_i}$. Specifically, we generate $\boldsymbol{x}_{ij} \in \mathbb{R}^d$ from a multivariate normal distribution, i.e., $\boldsymbol{x}_{ij} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{d\times d})$. Then, for individual i, the labels can be generated following $y_{ij} = \text{sgn}(\boldsymbol{\beta}_i^\top \begin{bmatrix} \boldsymbol{x}_{ij}, & 1 \end{bmatrix} + \epsilon_{ij})$ for $j = 1, \cdots, n_i$.

3.5.1 An overall evaluation

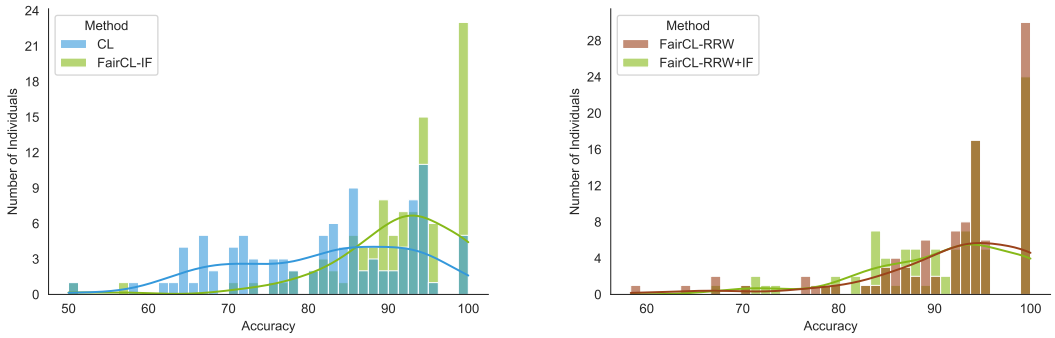
We compare the models that include IL, CL, FairCL-IF, FairCL-BIL, FairCL-RRW, and FairCL-RRW-IF in Figure 3.2 on a toy dataset generated under hierarchical strategy with

$d = 10$. Recall that IL is the learning strategy that learns each individual's model using the individual's data alone. As we can observe, CL significantly boosts the overall performances of the models than IL and also reduces the disparity among them. Its median accuracy is far higher than that of IL. Meanwhile, all the FairCL methods can further reduce the disparity among the performance of the individual models while achieving higher population-level average. We can also observe that if we have the knowledge on individual similarity, we may further improve the FairCL algorithms with their IF-regularized versions. For example, FairCL-RRW+IF seems better than FairCL-RRW, and FairCL-IF is clearly better than CL.

3.5.2 The impact of $\hat{K}$

In what follows we present more experimental results to show how the various hyperparameters impact the results. We first show how CL perform differently from IL for different $\hat{K}$s (recall that the true $K = 3$) under the hierarchical data generation strategy with $d = 10$ fixed. The histograms of the distributions of the model accuracy are shown in Figure 3.3. When $\hat{K}$=3 (which is the true number of canonical models), one can easily see that CL outperforms IL, as the entire distribution of the accuracy of the individual models of CL (blue) is more compact and both the mean and peak of the distribution are larger than the ones of IL (orange). CL also has fewer low-performance individual models and more high-performance individual models. As $\hat{K}$ increases to 5 (which is a little larger than the true value), CL still outperforms IL. However, when $\hat{K}$=10, CL seems to downgrade. The peak of CL shifts to the left of the peak of IL, and more low-performance individual models occur. This result shows that a proper selection of the parameter $\hat{K}$ is important for the success of CL, which is consistent with the observations in prior works [86]. In practice, this parameter can also be effectively identified using AIC/BIC and cross-validation [86].

The comparisons between FairCL-BIL and FairCL-RRW with CL are shown in Figure 3.7 and 3.8 (see Appendix 3.8.1.1). As we mentioned before, CL itself doesn't focus on the fairness issue, though it does lead to more fairness compared to IL. Compared with FairCL-BIL and FairCL-RRW, the performances of the individual models of CL are more disparate.



(a) CL v.s. FairCL-IF (b) FairCL-RRW v.s. FairCL-RRW+IF

Fig. 3.5: Comparison of test accuracy distributions on simulated data under a hierarchical strategy with prior knowledge on IF ($\hat{K}$=3, d=100).

We can also observe that the advantage of FairCL-BIL will not sustain if there is serious misspecification of the parameter $\hat{K}$, i.e., when $K = 10$, the gap between FairCL-BIL and CL seems to shrink and is no more significant. Different from FairCL-BIL, FairCL-RRW outperforms CL for all $\hat{K}$.

3.5.3 The impact of d

The above results show the performance of the models with $d = 10$. When d is increased to 100, the results are shown in Figures 3.4 (with $\hat{K}$=3). It can be seen that CL still outperforms IL. And the superiority of FairCL-BIL and FairCL-RRW over CL is even more significant than it is when $d = 10$. So is CL than IL. As the sample size of the individual data is relatively small given such a large d, it is foreseeable that IL couldn't perform well, but CL-based approaches really help the individuals model to learn collectively and reinforce each other.

3.5.4 The impact of IF regularization

To evaluate the effect of further incorporating individual similarity information into the CL-based models, we conduct more experiments and some results are shown in Appendix 3.8.1.2

(with $d = 10$). It can be observed in Figure 3.9(a) that the IF regularization can significantly improve CL's performance with the corresponding $\hat{K}$, even when we set $\hat{K} = 10$ that quite deviates from the true value $K = 3$. Now We fix $\hat{K}$ to this true value and explore the performance when d is changed. Obviously, regardless of d, FairCL-IF outperforms CL to a great extent as is shown in Figure 3.10(a) and Figure 3.5(a). For FairCL-RRW, IF regularization can also shift the distributions more towards the right side as shown in Figure 3.10(b). But the improvement may not be that significant, as FairCL-RRW has already shown great performance. IF regularization is not necessarily always useful in improving fairness. As [72] argues, the IF principle which requires similar treatment on similar individuals can't always guarantee fairness in the population. Sometimes, it may even degrade the performance of some FairCL methods. An example of such degradation can be seen in Figure 3.5(b).

3.5.5 In-depth analysis of the detailed experimental results

To reach a more insightful understanding of FairCL algorithms, we further examine some detailed statistics of the models' accuracy in Tables 3.1-3.3. In these tables, we not only present the mean test accuracy, but also the best 10% and the worst 10% test accuracy. FairCL-RRW can always beat IL, CL and FairCL-BIL in the worst 10% mean accuracy and the overall variance for whatever $\hat{K}$ (see the underlined results). We also include experiment results from additional methods: Group contribution matching (GCM) [249], Post-process for α-DP [250], and Subgroup Mixup (SGMixup) [251, 252]. These three methods represent three types of fair-aware machine learning approaches, i.e., matching, post-processing and data augmentation. From the tables, it can be observed that our FairCL-RRW still outperformed these methods.

When IF regularization is added, FairCL-IF improves upon CL and can usually achieve high mean performance. Similarly, FairCL-RRW+IF might also improve upon FairCL-RRW for some cases. But we can also observe that IF regularization doesn't necessarily shrink the variance as we compare the variances of FairCL-RRW+IF with FairCL-RRW, especially when model complexity increases. Looking at the statistics of different segments of the

Table 3.1: Statistics of the models' accuracy on simulated data under a hierarchical strategy (d=5).

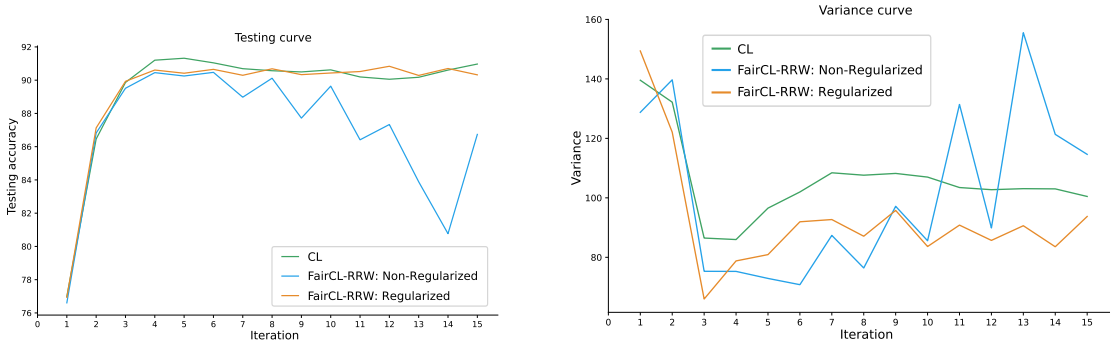
$\hat{K}$	**Methods**	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
	IL	88.56	100.00	64.28	113.57
	GCM	74.75	93.79	43.47	222.21
	Post-process	78.11	97.89	44.96	234.70
3	CL	93.91	100.00	73.48	84.28
	SGMixup+CL	93.51	100.00	74.46	72.24
	FairCL-BIL	94.74	100.00	76.42	67.68
	FairCL-RRW	95.99	100.00	80.18	48.30
	FairCL-IF	95.61	100.00	79.30	50.10
	FairCL-RRW+IF	**96.10**	100.00	**81.26**	**42.21**
5	CL	92.18	100.00	72.80	88.93
	SGMixup+CL	92.53	100.00	73.02	79.48
	FairCL-BIL	93.24	100.00	74.60	62.66
	FairCL-RRW	**95.92**	100.00	**80.20**	**47.54**
	FairCL-IF	95.08	100.00	77.43	57.86
	FairCL-RRW+IF	95.62	100.00	78.31	50.58
10	CL	89.86	100.00	67.99	92.25
	SGMixup+CL	88.76	100.00	68.48	105.71
	FairCL-BIL	90.74	100.00	70.44	88.84
	FairCL-RRW	**95.03**	100.00	**78.25**	**56.98**
	FairCL-IF	94.47	100.00	76.24	66.72
	FairCL-RRW+IF	94.85	100.00	77.76	59.86

Table 3.2: Statistics of the models' accuracy on simulated data under a hierarchical strategy (d=10).

$\hat{K}$	**Methods**	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
	IL	83.94	100.00	57.38	149.42
	GCM	71.55	93.95	44.69	192.24
	Post-process	77.82	96.46	56.96	132.17
3	CL	92.29	100.00	72.95	83.30
	SGMixup+CL	92.86	100.00	73.75	74.65
	FairCL-BIL	94.34	100.00	77.25	62.45
	FairCL-RRW	95.44	100.00	83.39	34.64
	FairCL-IF	95.12	100.00	84.17	32.75
	FairCL-RRW+IF	**96.38**	100.00	**85.62**	**25.27**
5	CL	91.94	100.00	69.78	89.65
	SGMixup+CL	92.03	100.00	70.67	81.64
	FairCL-BIL	93.66	100.00	76.01	68.44
	FairCL-RRW	**96.28**	100.00	**81.49**	**42.72**
	FairCL-IF	95.50	100.00	78.30	60.04
	FairCL-RRW+IF	95.61	100.00	79.81	52.30
10	CL	89.57	100.00	67.05	94.43
	SGMixup+CL	85.78	100.00	63.91	124.06
	FairCL-BIL	90.27	100.00	70.76	84.68
	FairCL-RRW	**94.54**	100.00	**78.75**	**49.40**
	FairCL-IF	93.56	100.00	71.73	78.73
	FairCL-RRW+IF	93.96	100.00	77.14	55.61

Table 3.3: Statistics of the models' accuracy on simulated data under a hierarchical strategy (d=100).

$\hat{K}$	**Methods**	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
	IL	63.17	84.00	38.44	161.63
	GCM	70.64	90.45	43.80	180.98
	Post-process	74.33	95.08	51.26	161.32
3	CL	88.26	100.00	65.26	113.35
	SGMixup+CL	86.41	100.00	65.73	108.10
	FairCL-BIL	89.01	100.00	70.48	87.98
	FairCL-RRW	**92.87**	100.00	**77.08**	**50.32**
	FairCL-IF	89.74	100.00	73.57	66.38
	FairCL-RRW +IF	90.35	100.00	73.89	59.29
5	CL	82.74	100.00	58.16	143.19
	SGMixup+CL	81.53	99.00	60.44	120.82
	FairCL-BIL	80.32	97.50	53.41	166.51
	FairCL-RRW	**89.11**	100.00	**70.17**	**89.41**
	FairCL-IF	85.80	100.00	58.51	157.89
	FairCL-RRW +IF	87.94	100.00	68.25	98.60
10	CL	75.76	96.04	47.22	196.72
	SGMixup+CL	74.66	91.55	43.83	190.85
	FairCL-BIL	74.87	94.49	50.27	160.43
	FairCL-RRW	**85.78**	100.00	**65.69**	**104.01**
	FairCL-IF	78.94	97.34	50.96	178.33
	FairCL-RRW +IF	84.39	100.00	58.47	151.50



(a) Mean accuracy of models on testing data

(b) Variance of accuracy of models

Fig. 3.6: The impact of TS regularization on FairCL-RRW.

population, we see that the way FairCL-RRW works is to enhance the performance of low-achieving individual models (i.e., evidenced by the higher mean accuracy of the worst 10%), which in turn further increases the mean accuracy of all the models, and reduces the variance. Note that these conclusions are drawn from experiments using the hierarchical strategy. Similar conclusions can be obtained if we simulate data using the totally random strategy (see Appendix 3.8.2).

3.5.6 The impact of TS regularization on computational stability

Figure 3.6 illustrates that the TS-regularized FairCL-RRW behaves more stably comparing with the non-regularized FairCL-RRW algorithm. It can be observed that the TS-regularized FairCL-RRW can converge quicker and maintains its stable performance over iterations with a good prediction performance and low variance among the individual models on the testing dataset. Note that this experiment is done on a simulated dataset generated under the totally random strategy with $K = 3, d = 10$, but overall this is a commonly observed phenomenon on other datasets in our experiments.

3.6 Real-world studies

We further evaluate the FairCL models using datasets collected from three different sectors including transportation, manufacturing and healthcare. Similarly, we compare the FairCL methods with not only IL and CL, but also three fairness-aware learning methods, i.e., GCM, Post-process and SGMixup-augmented CL.

3.6.1 Transportation demand management.

We started the real-world experiments from a real-world dataset collected from a personalized Transportation Demand Management (TDM) system [253, 254]. This TDM dataset studies travel behaviors in a reward system and consists of 828 participants while each participant has 13 observations. The number of attributes in each travel alternative is 4, which includes early schedule delay, late schedule delay, travel time saving, and reward points offered. And the participants make decisions on whether to accept the travel alternative given the rewards, which leads to binary responses. We randomly split each individual's data into training and testing sets by a ratio of 7:6. Following [86], we use 3-fold cross validation to determine the number of canonical models K using the training dataset.

Results of the different FairCL models are shown in Table 3.4. As the data dimension is small and the sample size is not large, although IL doesn't perform very well on average, and the mean accuracy of the worst 10% is quite low, it still achieves a mean accuracy over 70%. CL performs better than not only IL but also GCM and Post-process, and its performance can be improved by SGMixup, FairCL-BIL and FairCL-RRW. Particularly, FairCL-RRW reaches the smallest variance of accuracy while achieving a better mean performance. It is also easy to see that the mean accuracy of the worst 10% is much enhanced by FairCL-RRW compared to other methods. We further try to incorporate IL regularization, but since we don't have readily available prior knowledge of the individuals' similarity, here, following [86], the individual similarity is computed based on RBF kernel, i.e., the similarity between individual i and j is defined as $s_{ij} = \exp(-||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2/\sigma^2)$, where $\boldsymbol{x}$ represents the individual attributes. It can be observed that FairCL-IF can achieve smaller variance than FairCL-BIL

Table 3.4: Statistics of the models' accuracy on TDM dataset.

Methods	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
IL	72.36	100.00	30.38	498.37
GCM	64.77	100.00	16.11	605.85
Post-process	67.19	100.00	24.72	487.69
CL	74.32	100.00	35.33	454.33
SGMixup+CL	75.05	100.00	36.11	474.25
FairCL-BIL	75.17	100.00	38.33	413.06
FairCL-RRW	<u>76.92</u>	100.00	<u>42.22</u>	<u>412.91</u>
FairCL-IF	75.32	100.00	41.47	417.12
FairCL-RRW+IF	**77.76**	100.00	**42.69**	**395.92**

and higher mean accuracy of the overall population as well as the worst 10%, but is worse than FairCL-RRW. This is expected since the similarity information is not accurate enough.

3.6.2 Turbofan engine Degradation.

Another dataset we use in our numerical studies is the run-to-failure degradation trajectories of turbofan engines. The degradation simulation was carried out by the NASA Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dynamical model [255]. This dataset includes continuous observations of 90 different turbofan engines collected during multiple operational cycles before failure where each observation has 18 discriminative sensor measurements. And two failure modes including high-pressure compressor (HPC) degradation and fan degradation were considered in this dataset. We evaluate the ability of our methods to predict the failure of the engines indicated by the number of operation cycles. I.e., if an engine is operated more than 150 cycles, it is labeled as being at higher risk of failure. This gives rise to a binary classification task [256, 257].

We compare FairCL-BIL and FairCL-RRW with IL, GCM, Post-process, CL and SGMixup-

Table 3.5: Statistics of the models' performance on the CMAPSS dataset.

Methods	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
IL	77.61	100.00	15.78	784.57
GCM	82.55	100.00	28.48	563.01
Post-process	80.36	100.00	35.90	525.45
CL	83.56	100.00	36.23	525.67
SGMixup+CL	83.79	100.00	37.48	496.41
FairCL-BIL	84.38	100.00	37.91	511.72
FairCL-RRW	**86.34**	100.00	**42.39**	**417.07**

augmented CL. Due to the lack of prior knowledge, IF-regularized versions of our methods are not taken into consideration in the experiments. The results are reported in Table 3.5. Given the varying data sizes and data quality for different engines, IL can show extremely poor prediction performance on some individuals. This results in a low test accuracy on the worst 10%, indicating significant unfairness. GCM and Post-process significantly outperform IL on both accuracy and fairness, but they perform worse than CL. Though SGMixup can further shrink the variance which indicates better fairness, our FairCL-BIL and FairCL-RRW can improve the performance of CL to a larger extent. Particularly, FairCL-RRW demonstrates the highest mean accuracy while achieving the best performance fairness (highest worst 10% accuracy, lowest accuracy variance).

3.6.3 Surgical site infections.

We further test the methods using a dataset from healthcare that concerns surgical site infections (SSIs). SSI is one of the most common type of hospital-acquired infections which are usually associated with increasing morbidity, prolonged hospitalization, and higher mortality [258]. The heterogeneous conditions of patients and the high-stake situation call for personalized prediction models that can treat different patients fairly and give accurate diagnosis

Table 3.6: Statistics of the models' performance on the SSI dataset.

Methods	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
IL	74.27	100.00	22.37	645.17
GCM	91.34	100.00	57.18	231.32
Post-process	92.59	100.00	55.38	196.40
CL	92.94	100.00	59.29	190.65
SGMixup+CL	93.27	100.00	62.05	169.05
FairCL-BIL	92.95	100.00	63.17	162.83
FairCL-RRW	**98.14**	100.00	**81.44**	**59.06**

of SSI. Given this context, we conducted experiments on a SSI dataset which involves 259 patients whose wounds are continuously monitored in a range of 7-20 days. Each observation consists of 29 wound measurements, such as wound color, wound temperature and pain intensity, etc., together with a binary indicator showing the infection status marked by experts. Each individual's data is randomly splitted into training and testing sets with a ratio of 4:3. And similar to the experiments in the C-MAPSS dataset, we didn't consider IF regularized FairCL methods in comparison due to a lack of prior knowledge of the patients.

Table 3.6 shows our experimental results. While the performance of IL is poor, others perform much better with mean accuracy greater than 90%. Among these methods, FairCL-RRW achieves a mean accuracy much higher than all the other methods. When it comes to fairness, GCM, Post-process, and CL all outperform IL. And we can easily observe the significant improvement on the worst 10% accuracy by SGMixup, FairCL-BIL, and FairCL-RRW compared to CL. Compared to SGMixup, FairCL-BIL can improve the worst 10% accuracy and reduce the variance to a larger extent. And apparently, FairCL-RRW is superior than other methods as it improves the model accuracy of those low-achieving models and leads to a small variance. It can achieve the best performance on both accuracy and fairness.

3.7 Conclusion

Collaborative learning (CL) has been developed in the literature to learn personalized models for a heterogeneous population of individuals. In this work, we further develop the FairCL framework by incorporating fairness constraints into the CL framework and developing the corresponding computational algorithms. We further develop a self-adaptive FairCL formulation, named FairCL-RRW, that highlights a dynamic re-weighting approach to improve fairness of the individual models. We conduct theoretical analysis to reveal the connection of this novel re-weighting method with other FairCL formulations and the underlying rationale why it can reduce the variance of the performances of the individual models. Extensive numerical studies using both simulated and real-world datasets also show the advantages of the FairCL methods (particularly, FairCL-RRW) that can not only improve fairness but also the accuracy of the individual models.

3.8 Appendix

3.8.1 Additional results cited in Section 3.5

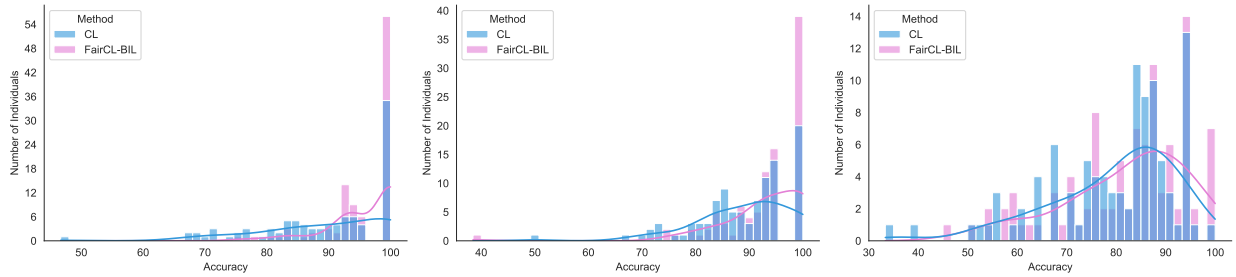
3.8.1.1 Test accuracy distributions of FairCL methods ($d = 10$)

For comparison with CL, the test accuracy distributions of our FairCL methods including FairCL-BIL and FairCL-RRW when $d = 10$ are shown in Figure 3.7 and 3.8.

3.8.1.2 The impact of IF

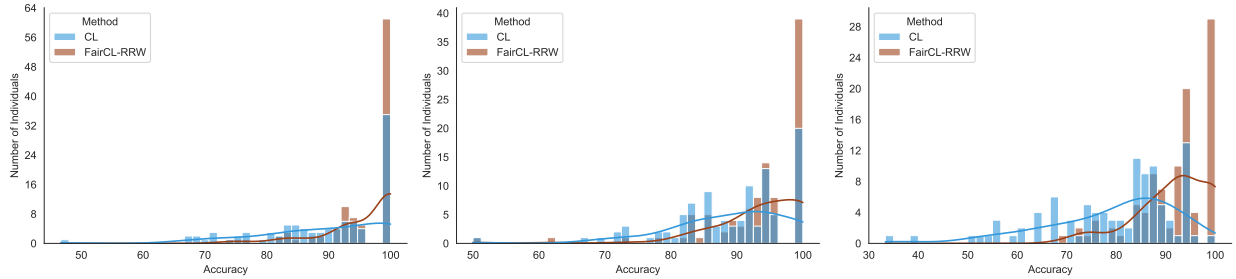
Figure 3.9 shows the comparison results between CL and FairCL-IF. With IF regularization which encodes the prior knowledge on individual similarity, the accuracy performance can be significantly improved.

For further investigation, we also show the comparison results of CL and FairCL-RRW with or without IF in Figure 3.10 when d is as small as 5 (with $\hat{K}$ fixed at the true value $\hat{K} = 3$). Compared to the results we report in the main text when $d = 100$, IF seems to slightly improve the performance of FairCL-RRW, though this impact doesn't stand out as it is on CL.



(a) $\hat{K}$=3 (b) $\hat{K}$=5 (c) $\hat{K}$=10

Fig. 3.7: Test accuracy distributions of FairCL-BIL on simulated data under a hierarchical strategy (d=10).



(a) $\hat{K}$=3 (b) $\hat{K}$=5 (c) $\hat{K}$=10

Fig. 3.8: Test accuracy distributions of FairCL-RRW on simulated data generated under a hierarchical strategy (d=10).

3.8.2 Additional experiments when using the totally random strategy for simulation

In this section, we further report the simulation results under the totally random strategy. Table 3.7-3.9 show the performance of IL, CL and the FairCL methods with various d values. Overall, we can observe the same results as the observation under the hierarchical strategy.

Table 3.7: Statistics of the test accuracy distribution on simulated data under the totally random strategy (d=5). The best values among all the methods have been highlighted and the best values among methods without prior knowledge on individual similarity have been underlined.

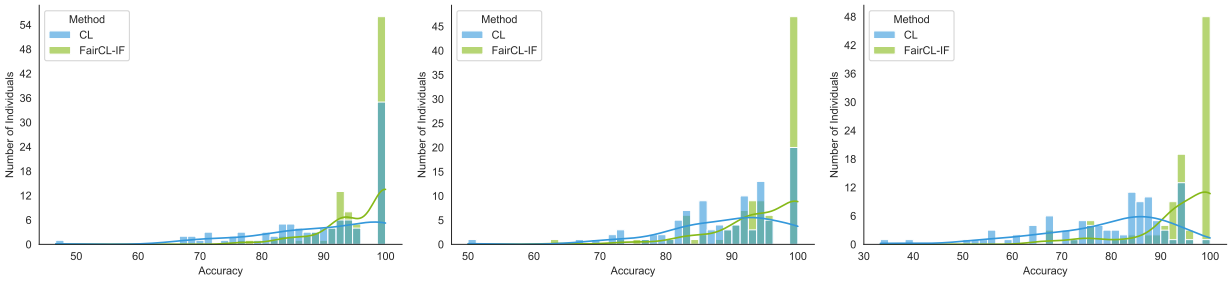
K	**Methods**	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
	IL	89.16	100.00	70.20	82.71
3	CL	92.96	100.00	73.72	74.78
	FairCL-BIL	93.09	100.00	74.58	67.42
	FairCL-RRW	93.93	100.00	**77.11**	**58.93**
	FairCL-IF	93.36	100.00	74.68	65.59
	FairCL-RRW +IF	**94.24**	100.00	75.46	64.43
5	CL	90.84	100.00	67.89	97.42
	FairCL-BIL	92.22	100.00	75.66	70.99
	FairCL-RRW	**93.86**	100.00	76.51	56.72
	FairCL-IF	93.03	100.00	74.70	61.42
	FairCL-RRW +IF	93.77	100.00	**77.33**	**51.07**
10	CL	89.71	100.00	65.44	104.86
	FairCL-BIL	92.46	100.00	73.42	66.74
	FairCL-RRW	92.59	100.00	73.71	67.61
	FairCL-IF	**93.10**	100.00	**74.82**	69.81
	FairCL-RRW +IF	92.85	100.00	72.79	**63.16**

Table 3.8: Statistics of the test accuracy distribution on simulated data under the totally random strategy (d=10). The best values among all the methods have been highlighted and the best values among methods without prior knowledge on individual similarity have been underlined.

K	**Methods**	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
	IL	85.08	100.00	64.18	112.81
3	CL	92.84	100.00	73.53	76.24
	FairCL-BIL	93.57	100.00	76.21	55.44
	FairCL-RRW	94.02	100.00	77.57	54.74
	FairCL-IF	94.08	100.00	77.67	59.76
	FairCL-RRW +IF	**94.16**	100.00	**78.12**	**52.21**
5	CL	90.13	100.00	70.68	94.82
	FairCL-BIL	91.92	100.00	72.13	83.85
	FairCL-RRW	**93.28**	100.00	**77.49**	**54.34**
	FairCL-IF	92.83	100.00	74.85	69.89
	FairCL-RRW +IF	92.10	100.00	75.80	64.94
10	CL	86.98	100.00	61.04	129.66
	FairCL-BIL	84.21	100.00	67.81	102.36
	FairCL-RRW	**92.65**	100.00	74.01	63.79
	FairCL-IF	90.76	100.00	70.57	96.33
	FairCL-RRW +IF	91.12	100.00	**75.02**	**62.58**

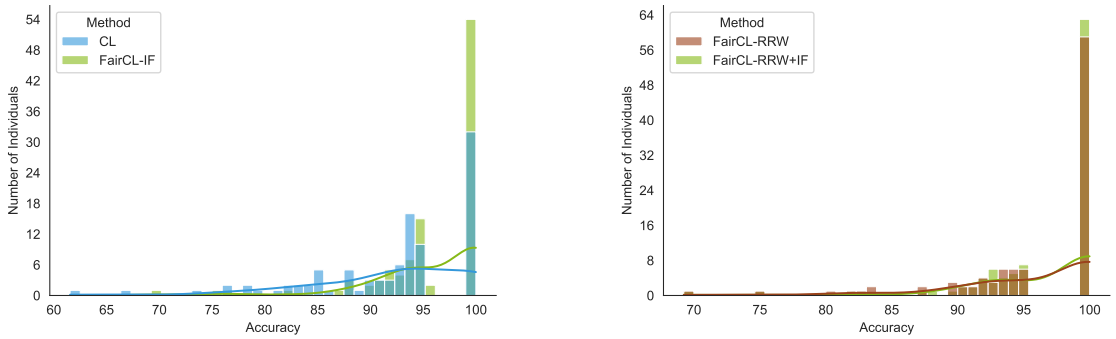
Table 3.9: Statistics of the test accuracy distribution on simulated data under the totally random strategy (d=100). The best values among all the methods have been highlighted and the best values among methods without prior knowledge on individual similarity have been underlined.

K	**Methods**	**Mean**	**Best 10%**	**Worst 10%**	**Variance**
	IL	63.14	83.50	38.81	161.05
3	CL	80.25	96.99	60.11	128.65
	FairCL-BIL	82.09	98.02	61.74	113.65
	FairCL-RRW	<u>86.67</u>	<u>100.00</u>	<u>69.15</u>	<u>82.59</u>
	FairCL-IF	**87.34**	**100.00**	69.28	80.91
	FairCL-RRW +IF	87.28	99.64	**71.91**	**74.87**
5	CL	78.55	95.75	59.31	133.60
	FairCL-BIL	80.27	96.53	58.03	140.45
	FairCL-RRW	<u>83.85</u>	<u>98.61</u>	<u>64.33</u>	<u>102.00</u>
	FairCL-IF	**84.57**	**100.00**	63.87	107.40
	FairCL-RRW +IF	84.45	97.93	**65.61**	**88.35**
10	CL	73.15	93.39	48.50	177.92
	FairCL-BIL	<u>79.22</u>	<u>97.04</u>	53.84	151.85
	FairCL-RRW	78.57	96.32	<u>54.52</u>	<u>148.66</u>
	FairCL-IF	77.79	97.01	53.42	157.03
	FairCL-RRW +IF	**80.67**	**97.54**	**57.30**	**139.94**



(a) $\hat{K}$=3 (b) $\hat{K}$=5 (c) $\hat{K}$=10

Fig. 3.9: Comparison of CL and FairCL-IF on simulated data under a hierarchical strategy (d=10).



(a) CL v.s. FairCL-IF (b) FairCL-RRW v.s. FairCL-RRW+IF

Fig. 3.10: Comparison of test accuracy distributions on simulated data under a hierarchical strategy with prior knowledge ($\hat{K}$=3, d=5).

3.8.3 Additional results on MNIST dataset

We also conducted experiments on the MNIST handwritten digit benchmark dataset [259] to show how FairCL-based methods can outperform IL and the basic CL. This dataset and its variants have been widely used in various machine learning tasks including model personalization [94, 260]. On this dataset, we consider a binary classification task of recognizing 0

(a) MNIST images. (b) Style reference. (c) Stylized images.

Fig. 3.11: An example of the style transfer.

for simplicity. If the digit is 0, it is labeled as 0. Otherwise, it is relabeled as 1. To build individual datasets, we consider $N = 30$ individuals and conduct two experiments. In the first experiment, for each individual i, we randomly pick up n_i images including both 0 and other digits as the training data, where n_i is an integer randomly selected between 7 and 17. We ensure that both labels will appear in the training dataset. And we randomly pick up 50 digit images as testing data. To further make the individual data more personalized (i.e., more heterogeneous), in the second experiment, for each individual, we create a personalized style by transforming the digit images into one calligraphy style using image style transfer. Here, for each individual, we randomly select a set of $0-9$ digits from a digit font dataset as style reference images and implement a VGG-19 convolutional neural network to realize the transfer following [261]. Figure 3.11 shows an example of the style transfer. This experiment task can be challenging due to the style differences and the potentially low quality of the digit images generated by the neural network. So we further create a disparity of sample sizes among the individuals, i.e., the sample size of each individual for model training varies between 5 and 40 while all the test sets consist of 50 digit images.

We compare FairCL-BIL and Fair-RRW with CL and IL. Due to the lack of prior knowl-

Table 3.10: Statistics of the models' accuracy on MNIST dataset (random).

Methods	**Mean**	**Best 30%**	**Worst 30%**	**Variance**
IL	82.07	90.60	72.60	81.20
CL	81.80	88.00	76.80	24.09
FairCL-BIL	85.53	90.60	80.60	27.49
FairCL-RRW	**90.07**	**94.00**	**85.80**	**14.53**

Table 3.11: Statistics of the models' accuracy on MNIST dataset (individual personalized).

Methods	**Mean**	**Best 30%**	**Worst 30%**	**Variance**
IL	80.33	93.00	65.20	204.56
CL	83.07	93.00	69.20	119.13
FairCL-BIL	88.60	**99.40**	73.80	148.84
FairCL-RRW	**90.00**	98.00	**80.00**	**65.87**

edge, IF-regularized versions of these models are not taken into consideration in the experiments. The results for both experiments are shown in Table 3.10-3.11. When the individual images are directly picked up from the full dataset (the first experiment), as Table 3.10 shows, the performance of IL is the worst due to the limited individual data. CL-based methods provide better performance in both mean accuracy and fairness. Among these methods, both FairCL-BIL and FairCL-RRW shrink the variance to a significant extent. Comparing with FairCL-BIL, FairCL-RRW offers more improvement in fairness. It can be observed that the mean performance is raised by more than 8% compared with CL. And it also significantly improves the worst 30% performance and shrinks the variance. Such results verify that FairCL-RRW leads to more accurate and fairer solution. In the second experiment, from Table 3.6, we can see the mean performance of IL gets lower than its performance in the previous experiment, which is not a surprise since the second experiment creates a more

challenging problem with style transfer. It should also be noted that the best 30% performance of all the methods can be higher provided with more training data while their worst 30% performance is downgraded a lot. Meanwhile, the variances are also enlarged given that the sample sizes of different individuals can be more varied. Among all the methods, IL is still largely beaten by CL-based methods. Compared to the basic CL, FairCL-BIL and FairCL-RRW both lead to much fairer solution by improving the worst 30% performance and shrinking the variance. Particularly, FairCL-RRW has increased the worst accuracy by more than 10%. In summary, the results of the two experiments further verify the superiority of the FairCL methods, especially FairCL-RRW, to improve fairness and also accuracy of the individual models.

3.8.4 Additional results on the impact of sample size

We performed additional experiments to test how the performance (in accuracy) of personalized models changes as sample size increases. We simulate 100 individuals, each has a set of data with 5 features and a binary label. And their sample sizes are determined randomly. Figure 3.12 shows the number of individuals for different sample sizes in this dataset.

Given this dataset, we ran IL, CL, FairCL-BIL and FairCL-RRW, respectively, and

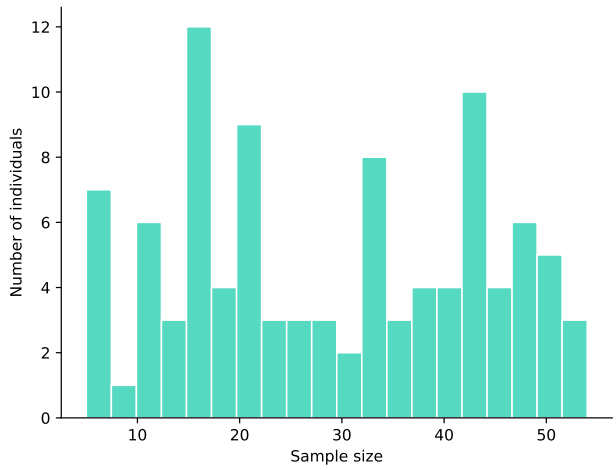
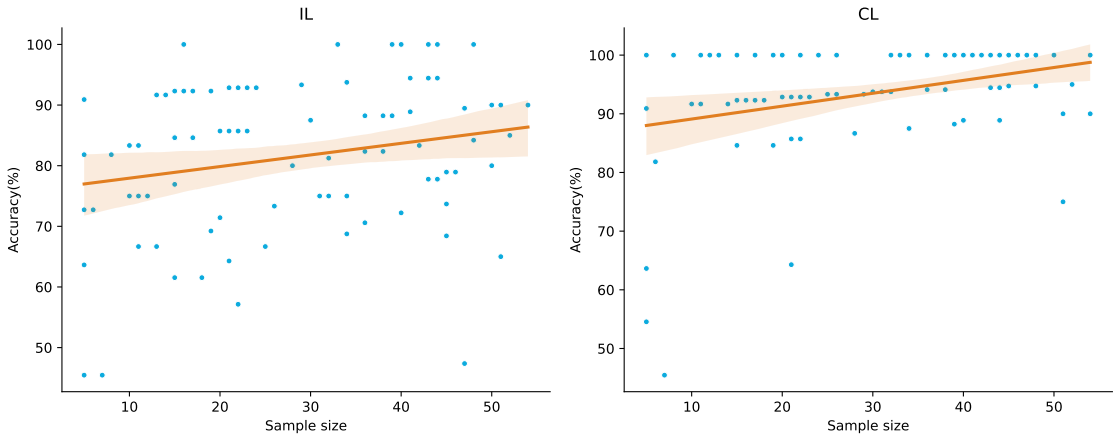
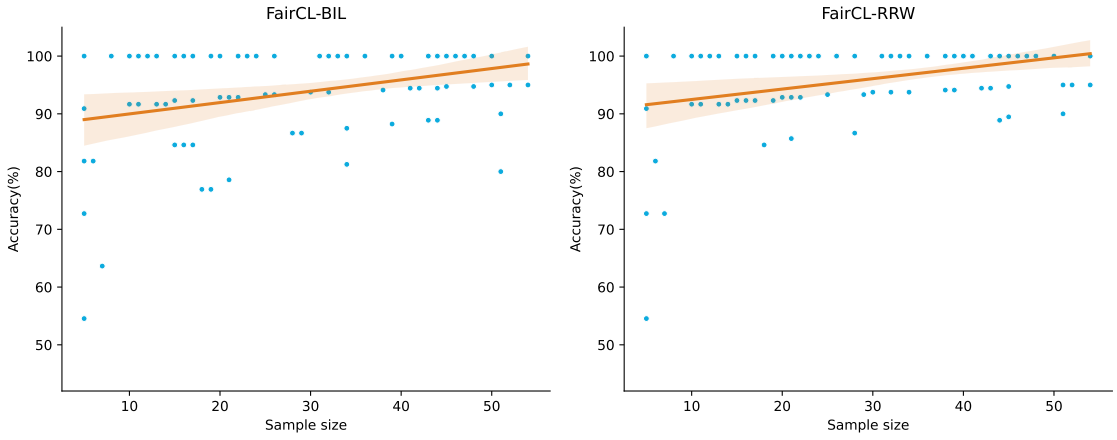


Fig. 3.12: The distribution of sample size of the 100 individuals.



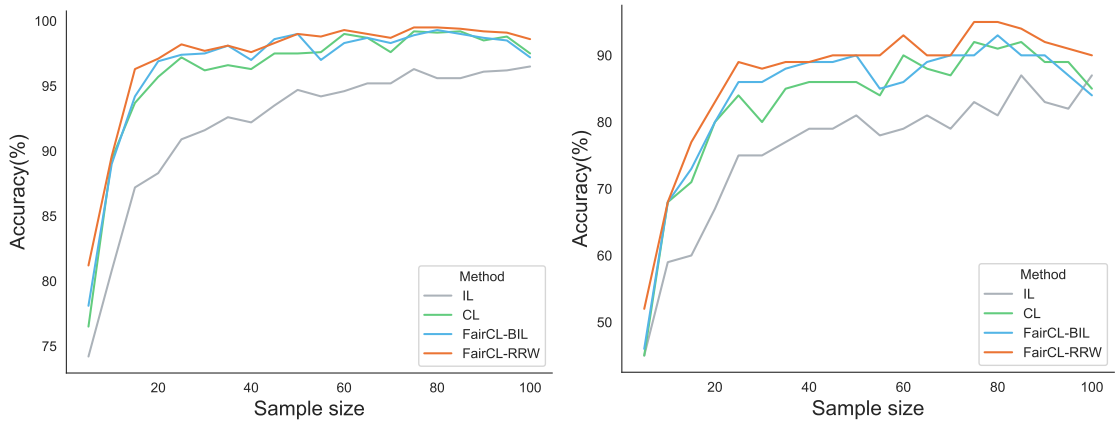
(a) IL (b) CL



(c) FairCL-BIL (d) FairCL-RRW

Fig. 3.13: An illustration of how the model performance changes as sample size increases.

demonstrate how the performance (in accuracy) of the personalized models changes with the sample size. Our results are shown in Figure 3.13. It is easy to see that, for all methods, the performance increases on average for the individuals with larger sample sizes, which demonstrates their advantage in learning. However, for IL, due to the disparities among the individuals, when the sample size increases, the accuracy can still vary wildly, while for CL-based methods, the impact of sample size on accuracy is much smaller, especially for



(a) Overall mean accuracy

(b) Worst 10% accuracy

Fig. 3.14: The impact of sample size on model performance.

FairCL-RRW. This also to some extent verifies their effectiveness in fairness improvement.

To further understand the impact of sample size on the model performance, we simulate another population of 100 individuals with data of the same sample size. Figure 3.14 demonstrates the changes of model performance as sample size increases on the simulated data. When data is extremely scarce for all the individuals, IL has a poor mean accuracy compared to CL-based methods. The proposed FairCL-RRW is significantly superior to other methods in both mean accuracy and worst 10% accuracy. When the sample size increases, all the methods start to perform better. And when the sample size is large enough for each individual, it is no surprise that IL can catch up with CL and even outperform CL in both mean accuracy and worst-case accuracy. But it should be mentioned that FairCL-RRW can outperform other methods regardless of the sample size and maintein its competitive performance even when the sample size increases.

Chapter 4

ROBUST LATENT DECISION THRESHOLD (R-LDT): MODELING USER CHOICE BEHAVIOR UNDER DATA CORRUPTION

Recent years have witnessed the emergence of many new mobile Apps and AI-enabled systems that are user-centered and actively interacting with users. These applications have been promising to address challenging societal problems such as congestion in transportation and behavior changes for healthier lifestyles in such an interactive way. User interactions with these social-technical systems not only hold great value to meet personalized needs but also are essential approaches to build trust between the users and the AI-enabled systems. It is no wonder that there has been an explosive growth of user data in many different areas [262], which has motivated much effort in modeling user behaviors in various applications. Although a common analytic formulation has been lacking, it is fair to say that there is a common goal that is to learn users' behavior patterns and understand their preferences so as to improve the system design to satisfy diverse user demands or make profits in a personalized and adaptive way [263, 264]. Still, there is a diverse range of definitions of user interactions and many interactions are somehow passive, e.g., in recommendation systems, users do not really interact with an intelligent agent that changes its behavior according to the interactions; but rather, all users deal with the same recommendation system and make their own choices such as a selection of items to purchase.

To better understand and serve users, some propose to involve active and stimulating interactions between users and systems [265, 266]. By meaningful user-system interactions, these social-technical systems can not only better understand user behavior patterns, but also influence their choice behaviors to serve collective goods, e.g., to stimulate consumers to

save water or ask people to wear masks for the control of the COVID-19 pandemic. Previous studies show that a reward system incorporating incentives can effectively encourage users to cooperate with the system by changing their behaviors [267, 268]. It has also been reported in [254] that with personalized interaction with users, their system is effective in changing user behaviors, i.e., the acceptance rate of the promoted suggestions reaches 68%, leading to significant travel time saving and congestion mitigation on the transportation system.

4.1 Introduction

In this chapter, we focus on this emerging application of user interaction that has not been well studied in the literature. That is, how users make choices when each choice is associated with a reward or incentive. A widely adopted family of models to model user's choice behaviors are discrete choice models (e.g., logistic regression) based on the Random Utility Maximization (RUM) theory [269]. However, as pointed out in [1], there is a fundamental limitation of these RUM-based models in our problem here since they assume that the data collection process is independent with the human behavior they aim to model. In other words, the process for data collection is assumed to not interfere with the human behavior to be observed and modeled. Actually, the RUM models were originally developed to explain population-level human behavioral tendencies, i.e., to understand the preferences of a population so that optimal economic or public policy could be made. Therefore, [1] proposed the Latent Decision Threshold (LDT) model that was the first model to characterize the user's choice making behavior when interacting with a reward-offering app. A distinct concept that separates LDT with the existing works such as the RUM models is the concept of "decision threshold", which is the tipping point where a user may change his/her decision between alternatives under a combined influence of the alternative's attributes, personal preferences, and reward. More details of LDT will be introduced in **Section 4.2**.

An important assumption implicitly made in [1] is the data the users generated is not corrupted. However, in the real world, it is expected that the user data should have a varying degree of quality, corrupted by noises, biases, and disruptions that are either caused

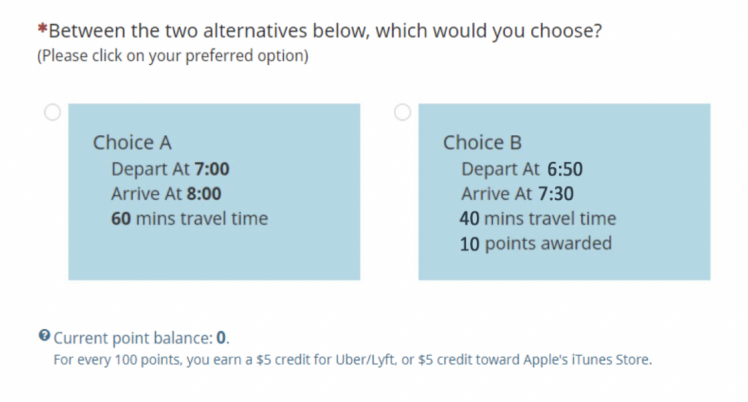


Fig. 4.1: An example of the TDM system in [1].

by random factors or adversarial motivations. This should not be a surprise due to the nature of the rewards as a double-edged sword. A reward system induces users to cooperate or change behaviors with rewards. This interaction loop could be a win-win strategy for the enhancement of both user experience and system design, but insider threats of "bad actors" can potentially lead to a deterioration of the system [270, 271]. And unfortunately, they do exist in many reward systems and hide among the normal users [272]. Even though the users might not be malicious users and have no adversarial motivations, it is human nature that they are usually attracted by the rewards and some might try to maximize their gains by cheating or other misconducts [273]. And some users may game or tease the system as a strategy for better profit. For this variety of reasons, the user data might be corrupted, and a robust formulation to learn the LDT model from the corrupted data is needed, which is our objective here.

4.1.1 A motivating example

To give a specific motivating example, here, we can use the example in [1]. A simple presentation is shown in Figure 4.1 that illustrates a typical scenario for the user to interact with the system, where choices are offered to the user with different rewards. The system was designed to learn the user's preferences on different attributes of the travel plans and

Table 4.1: The raw user data from [1].

Offer No.	1	2	3	4	5	6	7	8	9
Depart early (min)	30	10	0	30	0	60	0	30	0
Depart late (min)	0	0	60	0	10	0	60	0	10
Time save (min)	20	5	5	10	2	2	5	10	5
Reward (points)	10	20	20	10	30	30	90	50	50
Decision	1	1	1	-1	1	-1	-1	1	-1

derive optimal rewards to encourage users towards better travel behaviors. While majority of the users' data seems normal, there were some data corruptions that were observed. For example, Table 4.1 shows a small sample from the raw user data. A rough glance of the data could tell most of the data points make sense and are consistent with each other, but there are anomalies that result in a hard time in modeling. I.e., consider the offers No.3 and No.7 which provide the same choice but different rewards. Even though the reward is 90 for No.7 which is much greater than the reward given in offer No.3, the user rejected No.7. While this is one single example, we refer readers to more data examples in [1] that may give readers a deeper sense of the data corruption problem. If we do not enhance our statistical modeling framework with robust learning capacity, it is quite sure that these problematic data would mislead the learning process.

Beyond this example, we can see that the data corruption problem is universal. For example, the operation of health monitoring apps is largely dependent on the user data [274]. When the decision-making model of the app system is misled by corrupted data, it might give out wrong signals that risk the users' lives. Another example is the impact of data corruption on computer system security. For computer systems dependent on user-provided data, malicious attackers can perform data poisoning attacks even by a slight data corruption so that they can manipulate the system to steal user information [275]. Furthermore, in **Section 4.2.3**, we will further use some numerical examples to show how data corruption

can actually mislead the learning process and how we should address it. It is foreseeable that data corruption will emerge as a big research challenge that contains vast possibility and diversity of the forms, so at this stage of our research, we will not consider specific types of data corruption but aim to provide a general framework to enable robust learning. This general framework would ease some of the pains caused by data corruption in current practice and also provides a methodological baseline for future methods that can specifically address some well defined forms of data corruption.

4.1.2 Our proposed method and contribution

There have been many robust learning strategies developed in the literature. But which one works best in our problem is still unknown. We develop robust learning models using both L_1 norm and L_0 norm regularization and conducting a thorough evaluations of our models using both simulated and real-world data. Our contribution consists of the following aspects: **(1) Application-wise:** The emerging behavior-modifying apps will generate wider and deeper impact in many aspects of our daily life. As we are still at a very early stage in researching these behavior-modifying apps, our methodological study of robust learning strategy in this new problem context is timely and would contribute to the adaption of these new AI technologies in many areas. **(2) Methodology-wise:** We develop a robust learning extension called R-LDT using L_0 norm. We show the convergence of R-LDT to stationary points and discuss why L_0 norm is a better choice in our problem. Meanwhile, through experimental evaluations, we demonstrate that, among the existing robust learning frameworks, L_0 norm can outperform other methods in terms of accuracy and model estimation. And furthermore, we develop a user screening algorithm to identify potential bad actors.

4.2 The Latent Decision Threshold model and the challenge of data corruption

In this section, we present the formulation of the Latent Decision Threshold (LDT) model developed in [1] and discuss the challenge of data corruption to for LDT.

4.2.1 The formulation of LDT

As shown in Figure 4.2, the main innovation of LDT comparing with the RUM-based models is that the LDT model is specifically developed for modeling user choice behavior in user-centered reward-offering systems. The two models have a fundamental difference in conceptualizing the role of reward in this decision-making process, despite their common goal that is to learn user preferences from historical user data. Particularly, as shown in Figure 4.2, the RUM-based models mainly treat the attributes and the reward as the same without discrimination, while the LDT model is built on the concept of latent decision threshold. It is an individual-level function on alternatives that represents a tipping point at which users might change their mind on an alternative. Intuitively, an alternative has many explanatory attributes of interest, say, d attributes, that are denoted as $\boldsymbol{x} = (x_1, \cdots, x_d)$. The latent decision threshold function, denoted by $D(\boldsymbol{x})$, represented the combined force of the d attributes and the user preference $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_d)$. This latent decision threshold is formulated as a linear model of the attributes, $D(\boldsymbol{x}) = \boldsymbol{\beta}^\top \boldsymbol{x} = \beta_1 x_1 + \cdots + \beta_d x_d$. Here, β_i reflects the marginal cost of the ith attribute of an alternative. Since the reward is always positive, if β_i is positive, it means the corresponding attribute is a cost factor for the user.

As each alternative is associated with a reward, denoted as r, the decision-making rule in LDT of whether or not the user chooses this alternative is modelled as

$$y = \begin{cases} +1, & r \geq D(x) \\ -1, & r < D(x) \end{cases} \tag{4.1}$$

where y is the decision label, i.e., $y = 1$ represents acceptance while $y = -1$ represents rejection. Eq. (4.1) can be equivalently written as $y(r - D(x)) \geq 0$.

4.2.2 The max-margin learning formulation of LDT

Given n data points collected from a user, denoted as $\mathcal{D} = \{(\boldsymbol{x}_i, r_i, y_i)\}_i = 1, \cdots, n$, where $\boldsymbol{x}_i \in \mathbb{R}^p$, $r_i \in \mathbb{R}$, $y_i \in \{-1, 1\}$, the learning of LDT is to learn the model parameters $\boldsymbol{\beta}$ from

$\mathcal{D}$. In [1], a max-margin learning method is proposed for this learning task:

$$\begin{aligned} \min_{\boldsymbol{\beta},\boldsymbol{\xi}} \quad & \frac{1}{2}\boldsymbol{\beta}^\top\boldsymbol{\beta} + C\sum_{i=1}^{n}\xi_i \\ s.t. \quad & y_i(r_i - \boldsymbol{\beta}^\top\boldsymbol{x}_i) \geq 1-\xi_i,\ \xi_i \geq 0, \quad \forall j = 1,\cdots,n \end{aligned} \tag{4.2}$$

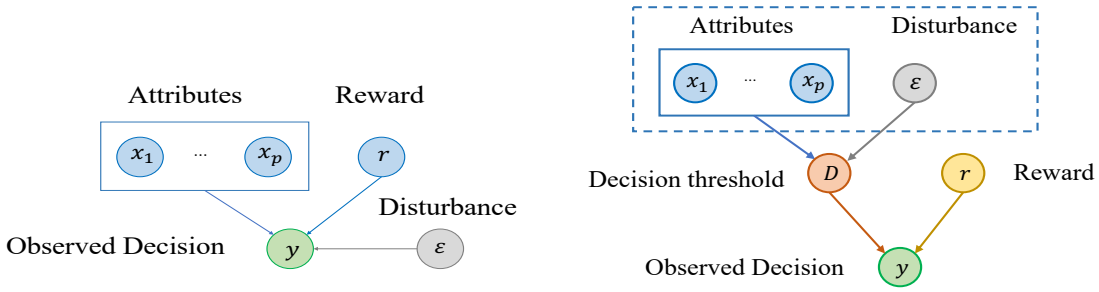
where C is a tuning hyperparameter that controls the penalty on soft margins $\xi_i (i = 1,\cdots,n)$. With the introduction of hinge loss, this minimization problem can also be equivalently rewritten as (with λ as a regularization parameter)

$$\min_{\boldsymbol{\beta}} \sum_{j}[1 - y_i(r_i - \boldsymbol{\beta}^\top\boldsymbol{x}_i)]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2. \tag{4.3}$$

This learning formulation looks similar to the formulation of soft-margin Support Vector Machine (SVM). The difference is that, in SVM, the bias parameter remains the same for all data points, while in this formulation, the bias parameter is essentially the reward r_i that changes across the data points. It is worthy of pointing out that the max-margin formulation of the learning is just one of the many approaches for parameter learning of the LDT model. We adopt this max-margin formulation here since it was shown to be effective in [1]. Going beyond the max-margin formulation is one of our research objectives in the future to fully unleash the conceptual potential of the LDT model in characterizing the user choice behaviors when interacting with choices with rewards.

4.2.3 Learning LDT under data corruption: The challenges

Data corruption problem is commonplace when collecting data from users [276]. In this subsection, we use specific examples to show how the learning problem of LDT could be severely disrupted by corrupted data points. Particularly, we consider a type of teasing behavior from users who may reject an alternative even when its reward exceeds their expectation, i.e., when $r \geq D(x)$. Users are prone to this type of behavior for understandable reasons, i.e., some users want to know how high the reward could go, to maximize their profit or simply satisfy some curiosity. Nonetheless, there could also be malicious users who want to cheat the system to earn more rewards or disrupt the system [277]. Either case, the collected data will



(a) RUM (b) LDT

Fig. 4.2: Illustration of differences between LDT with RUM with graphical representations

contain corrupted data points that come from these abnormal behaviors and therefore the learning algorithms can be misled [278]. For the learning of LDT, even with a max-margin formulation that allows a certain degree of data errors or model misclassification, such data corruptions can still be an issue.

Assume the true model is $D(x_1, x_2) = 0.4x_1 + 0.6x_2$, $y = \text{sgn}(r - D(x_1, x_2))$. Using this model, we randomly sample 6 data points, as shown in Table 4.4

The model parameters could be accurately learned by the max-margin learning algorithm proposed in [1]. Then we randomly corrupt just one data point in Table 4.4, i.e., by changing the label of the fifth sample from 1 to -1. This slightly modified dataset is shown in Table 4.2. Note that, in this sample, the reward $r = 1.6$ is higher than the true threshold $D(\boldsymbol{x}) = \beta_1 x_1 + \beta_2 x_2 = 0.6$, so the user should have accepted the reward. By reversing the label, we create an abnormal rejection. The estimated parameters by the LDT model is $\hat{\boldsymbol{\beta}} = (0, 1)$ which completely missed the importance of the variable x_1. Quantitatively, as the true parameters are $\boldsymbol{\beta} = (0.4, 0.6)$, we can also compute the mean square error (MSE), i.e., $\ell(\boldsymbol{\beta}, \hat{\boldsymbol{\beta}}) = \frac{1}{2}[(0 - 0.4)^2 + (1 - 0.6)^2] = 0.16$. Such a deviation cannot be ignored since it is quite comparable with the scale of $\boldsymbol{\beta}$.

Let us consider another example of data corruption, as shown in Table 4.3. It is created by adding an outlier $(x_1, x_2, r, y) = (6, 1, 59, -1)$ to the dataset in Table 4.4. For this corrupted

Table 4.2: A corrupted dataset, by reversing y of the 5th sample in Table 4.4.

Spl.	x_1	x_2	r	$D(\boldsymbol{x})$	y	y_{true}
1	1	1	2	1	1	1
2	2	2	3	2	1	1
3	1	1	0	1	-1	-1
4	2	2	1	2	-1	-1
5	0	1	1.6	0.6	-1	1
6	0	1	-0.4	0.6	-1	-1

Table 4.3: A corrupted dataset by adding a 7th sample into Table 4.4

Spl.	x_1	x_2	r	$D(\boldsymbol{x})$	y	y_{true}
1	1	1	2	1	1	1
2	2	2	3	2	1	1
3	1	1	0	1	-1	-1
4	2	2	1	2	-1	-1
5	0	1	1.6	0.6	1	1
6	0	1	-0.4	0.6	-1	-1
7	6	1	59	3	-1	1

sample, the reward is raised to an extremely high amount with a relatively large x_1, and the decision threshold $D(\boldsymbol{x}) = 3$. However, the reversed label $y = -1$ indicates the user still rejects the reward. We run the max-margin learning algorithm proposed in [1] and the estimated parameters are $\hat{\boldsymbol{\beta}} = (3, -1)$. Comparing with the true parameters $\boldsymbol{\beta} = (0.4, 0.6)$, the estimated parameters lead to a total opposite interpretation of the roles of the attributes in determining the user's conception of the choices.

The second example might suggest an impression that a corrupted sample with a higher reward can cause greater error in parameter learning for LDT. However, this is not always true. We conduct another case study. As shown in Table 4.5, the seventh sample with an enormously large reward is no more a corrupted sample. Instead, the corrupted sample is again the fifth sample whose label is reversed (the same as in Table 4.2). We run the max-margin learning algorithm proposed in [1] and the estimated parameters are $\boldsymbol{\beta} = (0, 1)$, which, again, quite deviate from the true parameters $\boldsymbol{\beta} = (0.4, 0.6)$.

One may notice that the sample size is pretty small in the examples. For user behavior modeling, many datasets in the real world do have small sample sizes, i.e., in [254] on

Table 4.4: The non-corrupted synthetic dataset.

Spl.	x_1	x_2	r	$D(\boldsymbol{x})$	y
1	1	1	2	1	1
2	2	2	3	2	1
3	1	1	0	1	-1
4	2	2	1	2	-1
5	0	1	1.6	0.6	1
6	0	1	-0.4	0.6	-1

average each user only had about 10 samples. This is because that the data collection in those applications is costly. This prospect may be improved in some applications such as video games or teaching systems. But still, collection of large user datasets in these systems requires long-term observations and we still need models and algorithms to learn their decision-making process at early stages of the system when user data is scarce.

One may also wonder why the slack variable in the max-margin formulation in (4.2) could not handle the problematic data points. This is actually an interesting point about LDT. It deserves a different kind of slack variable that should be placed not in the right hand side of the constraint of (4.2) but rather in the left and within the parenthesis. To see that, first, define $\Delta(r, \boldsymbol{x})$ as the profit margin an alternative could offer a user, i.e., $\Delta(r, \boldsymbol{x}) = r - \boldsymbol{\beta}^\top \boldsymbol{x}$. The data corruption problem here could not be simply addressed by the slack variable ξ_i because that, different from SVM, where the bias parameter remains the same so $\Delta(r, \boldsymbol{x})$ only changes with $\boldsymbol{x}$, here, it also changes according to r. As a consequence, learning LDT via the formulation (4.2) tends to be affected by those samples with large magnitudes of Δ, particularly when the Δs of them are inconsistent with their labels, the learning algorithm will try to fix the inconsistency not simply by increasing the slack variables ξ but also by pushing the model parameters towards a new direction. If an alternative (with attributes $\boldsymbol{x}$)

Table 4.5: Another example of a corrupted dataset.

Spl.	x_1	x_2	r	$D(\boldsymbol{x})$	y	y_{true}
1	1	1	2	1	1	1
2	2	2	3	2	1	1
3	1	1	0	1	-1	-1
4	2	2	1	2	-1	-1
5	0	1	1.6	0.6	-1	1
6	0	1	-0.4	0.6	-1	-1
7	6	1	59	3	1	1

and the associated reward r actually meets or exceeds the user's expectation, he/she should accept it by labeling it with $y = 1$. Thus, in ground truth we have $\Delta(r, \boldsymbol{x}) = r - \boldsymbol{\beta}^\top \boldsymbol{x} > 0$. But if the data is contaminated or this alternative is intentionally rejected by the user, we will have $y = -1$, but the profit margin Δ is unchanged. LDT will try to learn a $\hat{\boldsymbol{\beta}}$ that makes $\Delta(r, \boldsymbol{x}) = r - \hat{\boldsymbol{\beta}}^\top \boldsymbol{x} < 0$. Thus, the estimated $\hat{\boldsymbol{\beta}}$ must be quite different from $\boldsymbol{\beta}$.

Last but not least, while our proposed R-LDT has not been presented yet, it is worthy of mentioning that on these three data corruption case studies, the R-LDT could well recover the true parameters with proper selection of the tuning parameters. This observation aligns with the experimental results that will be shown in **Section 4.4**.

4.3 Robust learning for Latent Decision Threshold

4.3.1 The formulation of R-LDT

To mitigate the challenge as illustrated in **Section 4.2.3**, the idea of our R-LDT formulation is to find a set of non-corrupted samples and simultaneously learn a model $\boldsymbol{\beta}$ that fits the data well on this selected set. Specifically, our R-LDT is formulated as

$$\min_{\substack{\boldsymbol{\beta}, S \\ |S|=(1-\gamma)n}} \quad \sum_{i \in S} [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)]_+ + \frac{\lambda}{2} ||\boldsymbol{\beta}||_2^2. \tag{4.4}$$

This simple and direct scheme, however, is challenging to solve, as shown in some recent works [154, 153, 150] that also formulated their learning problems as a sample selection problem. The minimization problem (4.4) is non-convex jointly in $\boldsymbol{\beta}$ and S. But on the other hand, the knowledge of one set of parameter could make the optimization of another much easier. Still, the optimization of S is an integer programming problem challenging to solve, which motivates us to reformulate (4.4) as

$$\min_{\substack{\boldsymbol{\beta},\boldsymbol{b} \\ ||\boldsymbol{b}||_0 \leq \gamma n}} \quad \sum_{i=1}^{n} [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - b_i)]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 \tag{4.5}$$

The magnitude of b_i indicates how likely the data point i is corrupted. Here, notice that in reality there could be many specific reasons or scenarios that lead to the corruption of a data point, but the parameter b_i provides a general way for data quality evaluation that is useful for any scenario: any data point that could not be explained by the LDT mechanism that conforms to the majority of the data points is corrupted and should not be included in S. In other words, the complement of S should be the support set of $\boldsymbol{b}$. Thus, the sparsity of $\boldsymbol{b}$ can be controlled by the corruption ratio. To build such a connection, the L_0 norm of $\boldsymbol{b}$ is constrained. Different from L_1 or other norms, the L_0 norm provides a way to explicitly describe the potential data corruption. As a result, in consideration of such corruption, this reformulation brings in more robustness in learning. Based on it, we develop a computational procedure consisting of alternating updates of $\boldsymbol{\beta}$ and $\boldsymbol{b}$.

4.3.2 Optimization algorithms

Solving formulation (4.5) is not trivial due to its non-convexity. Algorithms that are based on the hard-thresholding technique have been discussed and provided with theoretical guarantee in [153], but they are under the settings of robust regression, which is a simpler case than LDT. In our LDT-based formulation (4.5), the max-margin formulation itself increases the complexity of the optimization of $\boldsymbol{b}$, not to mention the existence of the rewards $r_i (i = 1, \cdots, n)$. Put simply, the biggest challenge our formulation presents is that, there is no naturally defined "residuals" that are straightforward in regression to guide the optimization.

Algorithm 4.1 Alternating Updates of R-LDT

Input: Data $\mathbf{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]$, rewards $\boldsymbol{r} = [r_1, \cdots, \boldsymbol{r}_n]$, decisions $\boldsymbol{y} = [y_1, \cdots, y_n]$, the corruption parameter γ, tolerance ϵ, stopping time T

$\boldsymbol{\beta}^0 \leftarrow \mathbf{0}$, $\boldsymbol{b}^0 \leftarrow \mathbf{0}$, $S^0 \leftarrow \{1, 2, \cdots, n\}$, $t \leftarrow 0, k \leftarrow \gamma n$

while $||\boldsymbol{b}_{(S_t)}||_2 > \epsilon$ or $t < T$ **do**

  $\boldsymbol{\beta}^{t+1} \leftarrow \text{Update}(X, \boldsymbol{r}, \boldsymbol{y}, S_t, \boldsymbol{\beta}^t)$ // Algorithm 2

  $\boldsymbol{b}^{t+1} \leftarrow \text{Update}(X, \boldsymbol{r}, \boldsymbol{y}, \boldsymbol{\beta}^{t+1}, \boldsymbol{b}^t, S_t)$ // Algorithm 3

  $S_{t+1} \leftarrow \text{HardThreshold}(\boldsymbol{b}^{t+1}, n - k)$

  $t \leftarrow t + 1$

end while

$\boldsymbol{\beta} \leftarrow \text{Update}(X, \boldsymbol{r}, \boldsymbol{y}, \boldsymbol{b}^T, S^T)$

Output: $\boldsymbol{\beta}, \boldsymbol{b}, S$

To overcome this challenge, our optimization strategy is to update $\boldsymbol{\beta}$ and $\boldsymbol{b}$ with two separate algorithms, i.e., Algorithm 4.2 and Algorithm 4.3. And to update $\boldsymbol{b}$, as a surrogate concept to residuals, we update it based on the profit margin $\Delta(r, \boldsymbol{x}) = r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i$, which aims to heuristically ensure that the updated $\boldsymbol{b}$ can capture the inconsistency between $\boldsymbol{y}$ and the actual profit margin. **Theorem 4.3.1** and numerical experiments in **Section 4.5** and **Section 4.4** show that this design works effectively.

Specifically, Algorithm 4.1 shows how we alternately update the model $\boldsymbol{\beta}$, the corruption adjustment vector $\boldsymbol{b}$, and the non-corrupted set S. The algorithm starts with S being the full set, i.e, assuming there is no corruption. In each iteration of the algorithm, we first update the model $\boldsymbol{\beta}$, then update $\boldsymbol{b}$. And based on $\boldsymbol{b}$, we can finally update the set S. To obtain the latest estimation of S, we adopt a hard-thresholding strategy. The hard threshold function [153, 154] can be defined as:

$$\text{HardThreshold}(\boldsymbol{x}, k) = \{i_1, \cdots, i_k || x_{i_1}| \leq \cdots \leq |x_{i_k}| \leq \cdots \leq |x_{i_n}|\} \tag{4.6}$$

This hard-thresholding function ensures that the samples in the current active set S are

Algorithm 4.2 Update $\boldsymbol{\beta}$

Input: Attributes $\mathbf{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]$, rewards $\boldsymbol{r} = [r_1, \cdots, \boldsymbol{r}_n]$, decisions $\boldsymbol{y} = [y_1, \cdots, y_n]$, current active set S, current model $\boldsymbol{\beta}$

1: $\boldsymbol{\beta} \leftarrow LDT(\mathbf{X}_S, \boldsymbol{r}_S, \boldsymbol{y}_S, \boldsymbol{\beta})$

Output: $\boldsymbol{\beta}$

those with the smallest $|b_i|$ based on the latest LDT model. Therefore, its performance also depends on how we update $\boldsymbol{\beta}$ and $\boldsymbol{b}$, which are shown in Algorithm 4.2 and Algorithm 4.3, respectively. The intuition behind the above algorithms comes from an observation that LDT is an interesting combination of regression and classification, i.e., the sign of $\Delta(r, \boldsymbol{x})$ takes the form of a binary classification while the threshold takes the form of a linear regression model on the attributes $\boldsymbol{x}$. Thus, our proposed algorithms take advantage of this structure and will first learn a binary classification model to obtain $\boldsymbol{\beta}$, and then, use the learned parameters as the parameters of a regression model to derive the residuals $\boldsymbol{b}$, i.e., given $\boldsymbol{\beta}$ obtained in Algorithm 4.2 by solving (4.4) on the active set, $\boldsymbol{b}$ is updated in Algorithm 4.3 by bounding the residuals in the context of $y_i (i = 1, \cdots, n)$. In this way, Algorithms 4.1-4.3 can optimize over (4.4), or equivalently (4.5). In fact, we have the following result, which shows the Algorithms 4.1-4.3 could indeed effectively push $\boldsymbol{\beta}$ and S towards a potentially better solution through alternating optimization iteration by iteration.

Theorem 4.3.1. *Algorithm 4.1-4.3 ensures that the loss in* (4.4) *monotonically decreases until it converges to a stationary point.*

Proof. At each iteration, given $\boldsymbol{\beta}$, suppose there is a permutation $\sigma(1), \sigma(2), \cdots, \sigma(n)$ that sorts the corruptions $|b_1|, |b_2|, \cdots, |b_n|$ in an descending order, i.e.,

$$|b_{\sigma(1)}| \leq |b_{\sigma(2)}| \leq \cdots |b_{\sigma(n)}|.$$

In each iteration, the update of S in Algorithm 1, driven by hard thresholding, follows the rule to fill the set by elements with smallest $|b_i|$ as many as possible. Thus, we have

$$S_{t+1} = \{\sigma(1), \cdots, \sigma(m_t)\},$$

Algorithm 4.3 Update $\boldsymbol{b}$

Input: $\mathbf{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]$, $\boldsymbol{r} = [r_1, \cdots, \boldsymbol{r}_n]$, decisions $\boldsymbol{y} = [y_1, \cdots, y_n]$, current active set S, current model $\boldsymbol{\beta}$

1: **for** $i = 1, \cdots, n$ **do**
2: **if** $y_i = 1$ **then**
3: $b_i = \min\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\}$
4: **else**
5: $b_i = \max\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\}$
6: **end if**
7: **end for**
8: $\boldsymbol{b} \leftarrow [b_1, \cdots, b_n]$

Output: $\boldsymbol{b}$

where $m_t = |S_{t+1}| = (1-\gamma)n$. Note that the number of elements in the clean set $|S_t|$ at any iteration after the first iteration, i.e., $t \geq 2$, always keeps the same due to the hard thresholding operator, therefore, we have $m_{t+1} = m_t$.

Given **Lemma 4.8.1** in the Appendix, we have

$$\begin{aligned} L_{S_{t+1}}(\boldsymbol{\beta}) &= \sum_{i \in S_{t+1}} |b_i| + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 = \sum_{i=1}^{m_t} |b_{\sigma(i)}| + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 \\ &\leq \sum_{i \in S} |b_i| + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 = L_S(\boldsymbol{\beta}), \quad \forall S : |S| = m_t. \end{aligned}$$

which is followed by

$$S_{t+1} = \underset{S:|S|=(1-\gamma)n}{\arg\min} \; L_S(\boldsymbol{\beta}).$$

Note that $\boldsymbol{\beta}$ is always updated at the beginning of each iteration by Algorithm 4.2. When $\boldsymbol{\beta} = \hat{\boldsymbol{\beta}}_{t+1}$, we have

$$L_{S_{t+1}}(\hat{\boldsymbol{\beta}}_{t+1}) \leq L_{S_t}(\hat{\boldsymbol{\beta}}_{t+1}). \tag{4.7}$$

Meanwhile, the LDT optimization in Algorithm 2 indicates

$$\hat{\boldsymbol{\beta}}_{t+1} = \arg\min_{\boldsymbol{\beta}} L_{S_t}(\boldsymbol{\beta}),$$

which leads to

$$L_{S_t}(\hat{\boldsymbol{\beta}}_{t+1}) \leq L_{S_t}(\hat{\boldsymbol{\beta}}_t). \tag{4.8}$$

Therefore, combining (4.7) and (4.8), we can see the loss function monotonically decreases over iterations. □

4.3.3 Discussion: Why R-LDT adopts L_0 norm but not L_1 norm

As the use of L_0 norm in R-LDT dictates the considerable effort to develop an efficient optimization algorithm shown in **Section 4.3.2**, it is natural to wonder if R-LDT could adopt L_1 norm since it may lead to an easier optimization problem. Indeed, in a wide range of models from linear regression to classification, both L_0 norm and L_1 norm are adopted for sparse learning to achieve variable selection. And a general conclusion is that both norms have pros and cons, as L_0 norm provides an approximate solution to an exact problem (i.e., as variable selection is essentially a formulation that adopts L_0 norm), while the L_1 norm based approach can provide an exact solution to an approximated problem (i.e., as L_1 norm provides the best convex relaxation for the L_0 norm). In the problem of variable selection, indeed it is still an open question regarding which norm is better.

But as we dive deeper into our problem, trying to discern the pros and cons of each approach, a counter-intuitive and surprising result leads us to a very interesting discovery of the difference between the two norms. This difference is fundamental in our problem and context, as we will show in the following text. First, let's give a formal and precise definition of the alternative version of R-LDT using L_1 norm. To avoid confusion, we will call the following formulation L_1-LDT.

$$\min_{\boldsymbol{\beta},\boldsymbol{b}} \quad \sum_{i=1}^{n}[1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - b_i)]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 + \lambda_{L_1}||\boldsymbol{b}||_1 \tag{4.9}$$

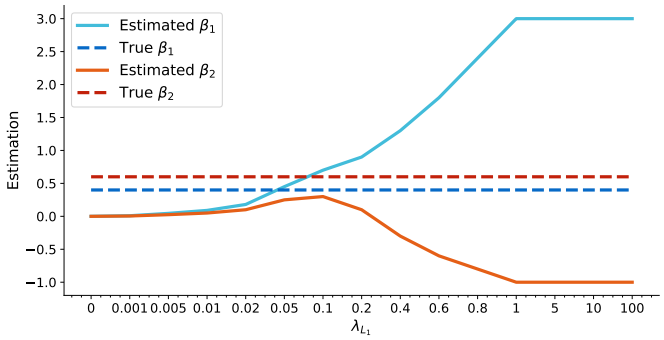


Fig. 4.3: The solution path of the estimated parameters in L_1-LDT.

We further implement the L_1-LDT on our synthetic examples in **Section 4.2** and the simulated data in **Section 4.5**, and later we will show that L_1-LDT performs worse than R-LDT in terms of both prediction accuracy and model parameter estimation. While this is not a surprise, a really interesting result is that, as shown in Figure 4.3, with the synthetic example in Table 4.3, we demonstrate the solution path of the estimated parameters of $\boldsymbol{\beta}$ by L_1-LDT according to the change of λ_{L_1}. Figure 4.3 gives us a full picture of the behavior of L_1-LDT so that we can understand better how it copes with the data corruption. It can be observed that, first, the estimation of $\boldsymbol{\beta}$ is simply $\mathbf{0}$ if we set $\lambda_{L_1} = 0$. This is the first contradiction we encounter, as we recall in LASSO, if we set the regularization parameter $\lambda = 0$, it would automatically degrade to a regular linear regression problem with only the least square loss function in effect. This contradiction disappears when we realize that if we set $\lambda_{L_1} = 0$, the parameters $\boldsymbol{b}$ are unconstrained, but $\boldsymbol{\beta}$ is still regularized by a L_2 norm, so $\boldsymbol{b}$ will take all the freedom to make the first term in (4.9) to be 0 and also push $\boldsymbol{\beta}$ to be 0

Next, as we increases λ_{L_1}, we can observe that the estimated parameters of $\boldsymbol{\beta}$ can never coincide with the true values. When λ_{L_1} is larger than 1, the estimated parameters $\hat{\boldsymbol{\beta}} = (3, -1)$ is the same as the parameter estimation given by LDT. To confirm that the phenomenon observed in Figure 4.3 is not just a singular case, we further analyze the same solution path of the estimated parameters of L_1-LDT on simulation data and we observe similar results

(see **Appendix 4.8.2**).

Therefore, the empirical evidences show that the L_1-LDT model shown in (4.9) does not work well for model parameter estimation of the LDT model. The solution path of the estimated parameters seem to consist of 3 stages, i.e., from left to right, there is a "dormant" stage where the estimated parameters are 0; then there is a "growth" stage where the estimated parameters change according to the change of λ_{L_1}, and the growth rate seems like linear; then there is a "degradation" stage where the estimated parameters remain the same (and actually, the same as the the estimated parameters of LDT) regardless of further increase of λ_{L_1}. If these three stages are a mathematical surety for L_1-LDT, then it means the L_1-LDT is not well poised for the parameter estimation task of LDT. Though, this is not a surprise, as L_1 norm is imposed on the parameter vector of $\boldsymbol{b}$ but not the model parameter vector $\boldsymbol{\beta}$. In other words, the L_1 norm here is not to directly help with the learning of $\boldsymbol{\beta}$. The only chance L_1-LDT can correctly estimate the model parameters is within the second stage, but at this stage the parameter estimation shows an artificial "growth" relation with the increase of λ_{L_1}. As the Figure 4.3 shows, even if one estimated parameter hits the true parameter, the other estimated parameters would be surely far off from their true values. With further in-depth analysis of the formulation of L_1-LDT that lead to the Theorem 4.3.2 and Theorem 4.3.3 shown in below, we conclude that the empirical observation of the behaviors of L_1-LDT as shown in Figure 4.3 is a mathematical certainty. As our problem is different from a general regression or classification problem, L_1 norm does not simply play a role as a convex relaxation of L_0 norm considering the interaction of the estimation of $\boldsymbol{\beta}$ and identification of $\boldsymbol{b}$. It actually causes a certain degree of incompatibility in L_1-LDT between the parameter estimation task and the corruption point detection task.

Theorem 4.3.2. *There exists a threshold $\lambda_u \leq 1$ such that when $\lambda_{L_1} \geq \lambda_u$, L_1-LDT will degrades to LDT.*

Proof. Note that the loss function of L_1-LDT is

$$\ell(\boldsymbol{\beta}, \boldsymbol{b}) = \sum_{i=1}^{n} [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - b_i)]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 + \lambda_{L_1}||\boldsymbol{b}||_1$$

$$= \sum_{i=1}^{n} [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) + y_i b_i]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 + \lambda_{L_1} \sum_{i=1}^{n} |b_i|$$
$$= \sum_{i=1}^{n} [[1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) + y_i b_i]_+ + \lambda_{L_1}|b_i|] + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2$$

Denote the ith loss function by $f(\boldsymbol{\beta}, b_i) = [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) + y_i b_i]_+ + \lambda_{L_1}|b_i|$. We fix $\boldsymbol{\beta}$ and divide the observations into two sets:

$$A_{\geq} = \{i : y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) \geq 1\}$$

$$A_{<} = \{i : y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) < 1\}$$

On one hand, on $A_{\geq}$, to minimize $\ell(\boldsymbol{\beta}, \boldsymbol{b})$, b_i should be 0. Otherwise, if $y_i b_i > 0$, by the monotonicity of the hinge loss, we have$f(\boldsymbol{\beta}, b_i) = [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) + y_i b_i]_+ + \lambda_{L_1}|b_i| \geq [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)]_+ + 0 = f(\boldsymbol{\beta}, 0)$, which indicates $b_i > 0$ will not leads to the minimum. On the other hand, if $y_i b_i < 0$, given that $1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) \leq 0$ on $A_{\geq}$, the hinge loss should be 0 as long as $y_i b_i < 0$, which results in

$$f(\boldsymbol{\beta}, b_i) = 0 + \lambda_{L_1}|b_i| = \lambda_{L_1}|y_i b_i| > 0 = f(\boldsymbol{\beta}, 0)$$

Again, $y_i b_i < 0$ doesn't lead to the minimum. Therefore, $y_i b_i = 0$. Since $y_i \neq 0$, we have $b_i = 0$.

On the other hand, for $A_{<}$, $y_i b_i \leq 0$ must hold. That is because, if $y_i b_i > 0$, we have $b_i \neq 0$. Then, again, we have $f(\boldsymbol{\beta}, b_i) = [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) + y_i b_i]_+ + \lambda_{L_1}|b_i| \geq [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)]_+ + \lambda_{L_1}|b_i| > f(\boldsymbol{\beta}, 0)$, which means $b_i \neq 0$ doesn't lead to the minimum. Thus, $y_i b_i \leq 0$, which easily gives $y_i b_i = -|b_i|$. Then we have

$$f(\boldsymbol{\beta}, b_i) = [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) - |b_i|]_+ + \lambda_{L_1}|b_i|$$

Let $C_i \triangleq 1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)$ and $z \triangleq |b_i|$. Then we can rewrite $f(\boldsymbol{\beta}, b_i)$ as

$$g(z) = [C_i - z]_+ + \lambda_{L_1} z$$

To minimize the loss function, we must have $0 \leq z^* \leq C_i$, where z^* is the optimal point. Otherwise, if $z^* > C_i$, we have $g(z^*) = \lambda_{L_1} z^* > \lambda_{L_1} C_i = g(C_i)$, which suggests C_i is better

than any $z^* > C_i$ for minimization. This contradicts to z^* is optimal. Since $z^* \leq C_i$, we have $C_i - z^* \geq 0$. And we can further have

$$g(z^*) = C_i - z^* + \lambda_{L_1} z^* = C_i + (\lambda_{L_1} - 1)z^*$$

When $\lambda_{L_1} \geq 1$, $g(z^*) \geq C_i = g(0) = f(\boldsymbol{\beta}, 0)$. So for minimization of $g(z)$, $z^* = |b_i^*| = 0$, which means L_1-LDT ignores all the potential corruptions and degrades to LDT. Thus, there must exists $\lambda_u \leq 1$, such that when $\lambda_{L_1} \geq \lambda_u$, L_1-LDT will degrade to LDT and $b_i^* = 0 (i = 1, \cdots, n)$. □

Theorem 4.3.3. *When $\lambda_{L_1} \leq \lambda_u$, L_1-LDT tends to update the estimated parameters with a locally linear change as λ_{L_1} changes.*

Proof. When $\boldsymbol{\beta}$ is fixed, based on the proof of Theorem 4.3.2, we see that for $i \in A_{\geq}$, $b_i = 0$. So we only need to consider $A_<$. Given that $z \leq C_i$, when $\lambda_{L_1} < 1$, $g(z)$ is monotonically decreasing with respect to z. Thus, z should be as large as possible to minimize the loss function, which leads to $z = C_i$. I.e., $|b_i| = 1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) = 1 - y_i r_i + y_i \boldsymbol{x}_i^\top \boldsymbol{\beta}, \forall i \in A_<$.

However, for optimization, we not only need to update $\boldsymbol{b}$, but also need to find the optimal $\boldsymbol{\beta}$. Obviously, to minimize the loss function, $\boldsymbol{\beta}$ tends to enlarge $A_{\geq}$ and shrink $A_<$ so as to reduce $f(\boldsymbol{\beta}, b_i)$ to the minimum 0. Thus, the set of $A_<$ and $A_{\geq}$ should rely on $\boldsymbol{\beta}$. If a $\boldsymbol{\beta}$ can ensure that $y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i) \geq 1$ holds for $\forall i = 1, \cdots, n$, which means $A_<$ is empty, the loss can be easily minimized. But in consideration of the existence of such $\boldsymbol{\beta}$ and the regularization on it, it might be infeasible, so $A_<$ is probably non-empty. To show the impact of λ_{L_1}, we substitute $|b_i|$ into $f(\boldsymbol{\beta}, b_i)$ and obtain

$$\ell(\boldsymbol{\beta}, b_i) = \sum_{i=1}^{n} f(\boldsymbol{\beta}, b_i) + \frac{2}{2}||\boldsymbol{\beta}||_2^2 = \lambda_{L_1} \sum_{i \in A_<(\boldsymbol{\beta})} (1 - y_i r_i + y_i \boldsymbol{x}_i^\top \boldsymbol{\beta}) + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2$$

which can be rewritten as a function of $\boldsymbol{\beta}$, i.e.,

$$h(\boldsymbol{\beta}) = \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 + \lambda_{L_1} \sum_{i \in A_<(\boldsymbol{\beta})} y_i \boldsymbol{x}_i^\top \boldsymbol{\beta} + \lambda_{L_1} \sum_{i \in A_<(\boldsymbol{\beta})} [1 - y_i r_i]$$

$$= \frac{\lambda}{2}\sum_{j=1}^{d}\beta_j^2 + \lambda_{L_1}\sum_{i\in A_<(\boldsymbol{\beta})} y_i\sum_{j=1}^{d} x_{ij}\beta_j + \lambda_{L_1}\sum_{i\in A_<(\boldsymbol{\beta})}[1-y_i r_i]$$
$$= \frac{\lambda}{2}\sum_{j=1}^{d}(\beta_j + \frac{\lambda_{L_1}}{\lambda}\sum_{i\in A_<(\boldsymbol{\beta})} y_i x_{ij})^2 + Const$$

Obviously, this is a quadratic function of β. To minimize the loss function, we should have

$$\beta_j^* = -\frac{\lambda_{L_1}}{\lambda}\sum_{i\in A_<(\boldsymbol{\beta})} y_i x_{ij}, \quad \forall j = 1,\cdots,d$$

Although this might not be reached due to the feasibility issue, we still see β_j should be close to $\eta_j(\lambda_{L_1}) \triangleq -\frac{\lambda_{L_1}}{\lambda}\sum_{i\in A_<(\boldsymbol{\beta})} y_i x_{ij} = \lambda_{L_1}\eta_j(j=1,\cdots,d)$ whose magnitude is linearly proportional to λ_{L_1}.

To see how $\boldsymbol{\beta}$ varies with λ_{L_1}, consider a possible division of the observations, $A_<$ and $A_\geq$. Within the corresponding feasible region S of $\boldsymbol{\beta}$, suppose the optimal solution that minimizes $h(\boldsymbol{\beta})$ is $\boldsymbol{\beta}^* = (\beta_1^*,\cdots,\beta_d^*)$. Denote $\boldsymbol{\eta}(\lambda_{L_1}) = (\eta_1(\lambda_{L_1}),\cdots,\eta_d(\lambda_{L_1}))$. Assume this solution is unique. Then, we have

$$\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}\in S} ||\boldsymbol{\beta} - \boldsymbol{\eta}(\lambda_{L_1})||_2 \tag{4.10}$$

It is easily observed that when $\lambda_{L_1} \to 0$, $\boldsymbol{\beta}^*$ should be very close to $\mathbf{0}$. Assume that λ_{L_1} increases with a very small amount Δ_λ. Correspondingly, the optimal point $\boldsymbol{\beta}^*$ should also be moved by a small change $\Delta_{\boldsymbol{\beta}^*} = (\Delta_{\beta_1},\cdots,\Delta_{\beta_d})$. Then, the new optimal point is $\boldsymbol{\beta}_\Delta^* = \boldsymbol{\beta}^* + \Delta_{\boldsymbol{\beta}^*}$ which satisfies

$$\boldsymbol{\beta}_\Delta^* = \arg\min_{\Delta_{\boldsymbol{\beta}^*}\in\Delta S} ||\boldsymbol{\beta}^* + \Delta_{\boldsymbol{\beta}^*} - \boldsymbol{\eta}(\lambda_{L_1}+\Delta_\lambda)||_2$$

where $\Delta S = \{\boldsymbol{\beta} - \boldsymbol{\beta}^* | \boldsymbol{\beta}\in S\}$. By triangle inequality, we have

$$\begin{aligned} ||\boldsymbol{\beta}^* + \Delta_{\boldsymbol{\beta}^*} - \boldsymbol{\eta}(\lambda_{L_1}+\Delta_\lambda)||_2 &= ||\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1}) + \Delta_{\boldsymbol{\beta}^*} - \boldsymbol{\eta}(\Delta_\lambda)||_2 \\ &= ||\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1}) - (\boldsymbol{\eta}(\Delta_\lambda) - \Delta_{\boldsymbol{\beta}^*})||_2 \\ &\geq |||\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1})||_2 - ||\boldsymbol{\eta}(\Delta_\lambda) - \Delta_{\boldsymbol{\beta}^*}||_2| \end{aligned} \tag{4.11}$$

Given $\boldsymbol{\beta}^*$ and λ_{L_1}, the equality holds when $\Delta_{\boldsymbol{\beta}^*} = \boldsymbol{\eta}(\Delta_\lambda) - c(\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1})), c \in \mathbb{R}$. It is clear that $\Delta^*_{\boldsymbol{\beta}}$ is a linear function of Δ_λ with λ_{L_1} fixed to reach the minimum. By denoting $\delta(\lambda_{L_1}) = ||\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1})||_2$ and $K = \sqrt{\sum_{j=1}^{d}(\sum_{i\in A_<} y_i x_{ij})^2}$, we notice that

$$||\Delta_{\boldsymbol{\beta}^*}||_2 = ||\boldsymbol{\eta}(\Delta_\lambda) - c(\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1}))|| \leq ||\boldsymbol{\eta}(\Delta_\lambda)||_2 + c||(\boldsymbol{\beta}^*) - \boldsymbol{\eta}(\lambda_{L_1})||_2 \leq \frac{K\Delta_\lambda}{\lambda} + c\delta(\lambda_{L_1})$$

When $\delta(\lambda_{L_1}) = 0$, obviously we have $||\Delta_{\boldsymbol{\beta}^*}||_2 \leq \frac{K\Delta_\lambda}{\lambda}$. By **Lemma 4.8.2**, we know that there exists a bound ϵ^* for $\boldsymbol{\beta}^*$ such that when $||\Delta_{\boldsymbol{\beta}^*}||_2 \leq \epsilon^*$, $\boldsymbol{\beta}^*_\Delta \in S$. Since Δ_λ can be arbitrarily small, we let $\Delta_\lambda \leq \frac{\lambda\epsilon^*}{K}$ that ensures $\boldsymbol{\beta}^*_\Delta \in S$. At this time, $\Delta_{\boldsymbol{\beta}^*} = \boldsymbol{\eta}(\Delta_\lambda)$, i.e., $\Delta_{\beta^*_j} = \lambda_{L_1}\eta_j$. It indicates the estimated $\boldsymbol{\beta}$ changes linearly as λ_{L_1} changes in a small local region. It should be mentioned that, when λ_{L_1} is so small that it approximates 0, we can easily observe $\boldsymbol{\beta}^* \to \mathbf{0}$ (i.e., as shown in Figure 4.10). Thus, $\delta(\lambda_{L_1}) \to 0$. In this case, there is surely a linear relationship between $\beta^*_j (j = 1, \cdots, d)$ and λ_{L_1}.

When $\delta(\lambda_{L_1}) > 0$, we notice that, to reach the minimum, the right-hand side of (4.11) should be

$$|||\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1})||_2 - ||\boldsymbol{\eta}(\Delta_\lambda) - \Delta_{\boldsymbol{\beta}^*}||_2| = |1 - c| \cdot ||\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1})||_2 = |1 - c|\delta(\lambda_{L_1}) > 0$$

However, due to the constraint $\Delta_{\boldsymbol{\beta}} \in \Delta_S$, this minimum may not be reached if c is not appropriately set. To determine the value of c, we first notice the constraint $\boldsymbol{\beta}^*_\Delta = \boldsymbol{\beta}^* + \Delta_{\boldsymbol{\beta}^*} \in S$. By the definition of $\boldsymbol{\beta}^*$, it gives

$$\delta(\lambda_{L_1}) = ||\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1})||_2 \leq \min_{\Delta_{\boldsymbol{\beta}^*}} ||\boldsymbol{\beta}^* + \Delta_{\boldsymbol{\beta}^*} - \boldsymbol{\eta}(\lambda_{L_1})||_2 = |1 - c|\delta(\lambda_{L_1})$$

It is easy to see that $|1 - c| \geq 1$, which leads to $c \in (-\infty, 0] \cup [2, \infty)$. But meanwhile, we also note that $||\boldsymbol{\beta}^* + \Delta_{\boldsymbol{\beta}^*} - \boldsymbol{\eta}(\lambda_{L_1} + \Delta_\lambda)||_2$ is minimized when c is as close to 1 as possible. Thus, we should have $c = 0$ or $c = 2$. However, $c = 2$ cannot ensure $\Delta_{\boldsymbol{\beta}^*} \in \Delta_S$. If $c = 2$, we have

$$\boldsymbol{\beta}^*_\Delta = \boldsymbol{\beta}^* + \Delta_{\boldsymbol{\beta}^*} = (1 - c)\boldsymbol{\beta}^* + c\boldsymbol{\eta}(\boldsymbol{\lambda_{L_1}}) + \boldsymbol{\eta}(\Delta_\lambda) = -\boldsymbol{\beta}^* + 2\boldsymbol{\eta}(\boldsymbol{\lambda_{L_1}}) + \boldsymbol{\eta}(\Delta_\lambda)$$

Suppose this $\boldsymbol{\beta}^*_\Delta \in S$. By **Lemma 4.8.2**, when Δ_λ is very small, we should also have $-\boldsymbol{\beta}^* + 2\boldsymbol{\eta}(\lambda_{L_1}) \in S$. Denote it by $\boldsymbol{\beta}' = -\boldsymbol{\beta}^* + 2\boldsymbol{\eta}(\lambda_{L_1})$. Then we have

$$||\boldsymbol{\beta}' - \boldsymbol{\eta}(\lambda_{L_1})||_2 = || - \boldsymbol{\beta}^* + \boldsymbol{\eta}(\lambda_{L_1})||_2 = ||\boldsymbol{\beta}^* - \boldsymbol{\eta}(\lambda_{L_1})||_2 = \min_{\boldsymbol{\beta}} ||\boldsymbol{\beta} - \boldsymbol{\eta}(\lambda_{L_1})||_2$$

which suggests $\boldsymbol{\beta}' \in S$ is also the optimal solution to (4.10). This contradicts with the unique optimality of $\boldsymbol{\beta}^*$. Therefore, we have $c = 0$, which results in $\Delta_{\boldsymbol{\beta}^*} = \boldsymbol{\eta}(\Delta_\lambda)$.

In conclusion, regardless of $\delta(\lambda_{L_1})$, in L_1-LDT, the change of $\boldsymbol{\beta}$ is locally linear on each dimension as λ_{L_1} changes. □

Theorem 4.3.2 explains the "degradation" stage while Theorem 4.3.3 explains the "growth" stage observed in Figure 4.3. Intuitively, in our context, L_0 and L_1 norm entice two different operations in the parameter estimation process. L_0 norm leads to a two-staged process in R-LDT. It first identifies the corrupted points, i.e., non-zero elements in $\boldsymbol{b}$, and after that, the magnitude of the corruption points does not really impact the estimation of $\boldsymbol{\beta}$ as no penalty is imposed on the magnitude of the corrupted points. But for L_1 norm, it entails a one-stage process in L_1-LDT that directly causes conflict between the estimation of $\boldsymbol{b}$ with the estimation of $\boldsymbol{\beta}$, because the use of L_1 norm means to allocate magnitudes to all elements of $\boldsymbol{b}$ rather than only a selected few. For small λ_{L_1}, L_1-LDT tends to give more weight on $\boldsymbol{b}$ than $\boldsymbol{\beta}$. But for large λ_{L_1}, L_1-LDT tends to ignore $\boldsymbol{b}$ but only focuses on the estimation of $\boldsymbol{\beta}$, and thus degrades to the model of LDT. On the other hand, R-LDT is more adaptive despite no guarantee of global optimum convergence. As [279] pointed out, for good performance one does not always need to find the global optimum. On the contrary, in practice, finding good local optimums in a robust, reliable, and relatively easy way is more important.

For a comparison with L_1-LDT, we also show the path trajectories of the estimated parameters given by R-LDT with the same synthetic example in Table 4.3. It can be seen from Figure 4.4, in accordance with our analysis, R-LDT can approach the true parameters far better than L_1-LDT.

In the following sections, we will further compare R-LDT with L_1-LDT and show R-LDT will achieve better performance on both simulated and real-world data. Last but not least,

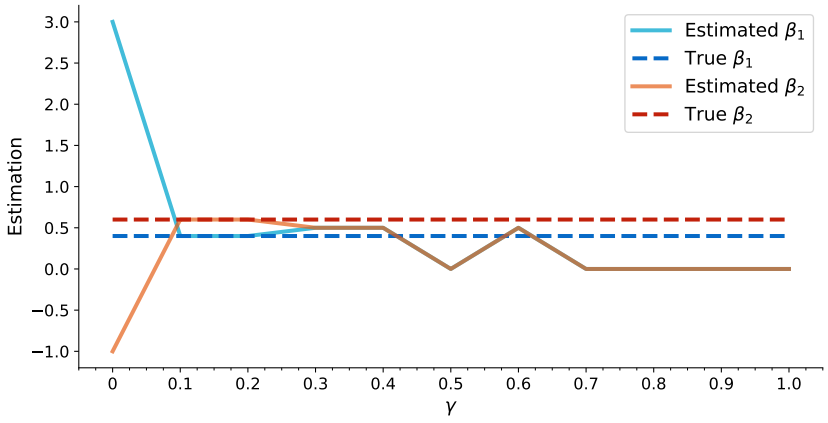


Fig. 4.4: The solution path of R-LDT on the synthetic example.

while based on R-LDT we can readily derive algorithms for bad actor detection since it exactly identifies corrupted data points, it is not that easy for L_1-LDT and other similar approximation-based methods along the line to achieve this.

4.4 Simulation Studies

4.4.1 Data generation.

The simulation is conducted under a two-step data-generating mechanism. First, we generate the baseline data from a true model. Then, the data is corrupted following a corruption mechanism. Suppose there are n samples to be generated, while each sample is a triple that consists of attributes, rewards and labels. The attributes $\boldsymbol{x_i}(i = 1, \cdots, n)$ are sampled from a $d-$dimensional normal distribution as $\boldsymbol{x}_i \overset{i.i.d}{\sim} N(\mathbf{0}, \mathbf{I}_d)$, here, for simplicity, $\mathbf{I}_d$ is a $d \times d$ identity matrix. The generation of the labels $y_i(i = 1, \cdots, n)$ follows the decision rule (4.1), i.e., $y = \text{sgn}(r - \boldsymbol{\beta} \cdot \boldsymbol{x})$. Rewards $r_i(i = 1, \cdots, n)$ are generated by the three mechanisms as described in [1]: **(1) Random reward.** The rewards are generated randomly without many predefined mechanism. In our simulation studies, they are generated by $r_i \overset{i.i.d}{\sim} N(0, 2)$. **(2) Contribution reward.** This award mechanism assumes the system has set the rewards based on its own rules and costs. To simulate such rewards, we first simulate

reference coefficients predefined by the system by generating a system-level preference vector $\boldsymbol{\beta} \sim N(\mathbf{0}, \mathbf{I}_d)$ and normalize it by L_2-normalization, i.e., $\beta_j := \beta_j / ||\boldsymbol{\beta}||_2$ for $j = 1, \cdots, d$. Then we generate r_i by $r_i = \boldsymbol{\beta} \cdot \boldsymbol{x}_i + \epsilon$, where $\epsilon_i \sim U(-1, 1)$. **(3) Predictive reward.** When the system has adequate knowledge about a user' preferences $\boldsymbol{\beta}$, the system can design rewards close to the users' decision thresholds. For this case, we generate r_i by $r_i = \boldsymbol{\beta}^\top \boldsymbol{x}_i + \epsilon$, where $\epsilon_i \sim U(-1, 1)$.

We then introduce corruption on this baseline data. Without loss of generality, in our simulation studies, we focus on the behavior of users who reject alternatives that offer rewards that exceed their expectations. The system usually continues to offer the same alternative with higher rewards but the user will continue to reject in a hope to gain a better offer with higher reward. To mimic this data corruption, a uniform corruption is added to the reward, i.e., $\tilde{r} = r + \delta_b$, where $\delta_b \sim U(10, 20)$, and the labels of these samples are reversed. Note that we only consider the case that users reject alternatives with rewards higher than their expectation. It is less usual that users will accept alternatives with rewards less than their expectation. Thus, in this study, we only pick those positive samples (i.e., the true labels are 1) for data corruption. The corruption ratio α refers to the ratio of positive samples that are corrupted in the dataset.

4.4.2 Experiment setup.

We generate multiple datasets with various settings of three parameters: the sample size of the training data n, the data dimension d, and the corruption ratio of the positive α. Ranges of the parameters are: $\alpha \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7\}$, $n \in \{10, 20, 30, 50, 100\}$ and $d \in \{2, 5, 10\}$. For each baseline dataset, we randomly split the dataset into training and test sets with a 1:1 split ratio. The training data is further corrupted following method outlined in **Section 4.4.1**. We use the corrupted training data to train the models and use the uncorrupted testing data to derive fair evaluation of the model performance. To train the models, we use cross-validation to set the parameters of the models. To compare the performance of LDT and R-LDT, we evaluate their prediction accuracy on the testing

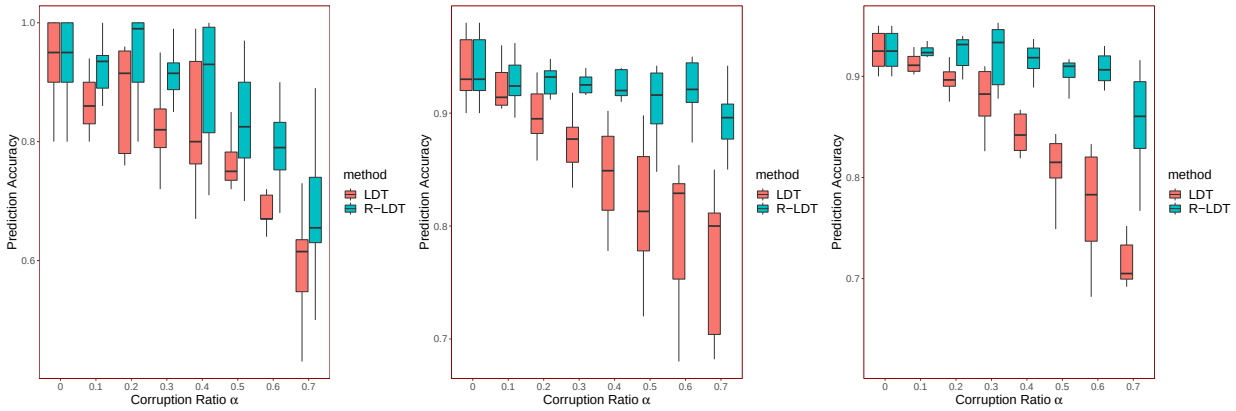
dataset. It measures how many samples in the test set of which they can correctly predict the labels. Another metric we use is the mean L_2 error, i.e., $||\boldsymbol{\beta}-\hat{\boldsymbol{\beta}}||_2^2/d$. This error measures how closely the estimator $\hat{\boldsymbol{\beta}}$ estimate the true $\boldsymbol{\beta}$.

4.4.3 Results: R-LDT enables robust learning of LDT

Figures 4.5 - 4.7 present the boxplots for prediction accuracy of LDT and R-LDT on the testing datasets under different settings of rewards. From these plots, we can see that, regardless of the sample size n or data dimension d, when the corruption ratio α increases, the prediction accuracy decreases for both LDT and R-LDT. The trend occurs under all settings, and it is easier to be seen when d and n are large. However, compared to LDT, the performance of R-LDT deteriorates at a much lower speed. For different corruption ratios, the accuracy of R-LDT looks better than LDT. And the gap between them becomes larger when α grows. This indicates that R-LDT is more resilient than LDT to data corruption. We also notice that when α is large, i.e., as large as 0.7, which means most of the positive samples have been corrupted, the prediction accuracy of LDT can even fall down below 0.6 in the worst case, which is just slightly better than random guess. However, the prediction accuracy of R-LDT can achieve about 0.8 on average, and is around 0.7 even in the worst case. In summary, R-LDT is a more robust model than LDT in terms of prediction accuracy. A similar observation can be made based on parameter estimation as well. We refer readers to see the mean square error (MSE) of LDT and R-LDT under different settings of rewards in **Appendix 4.8.4**.

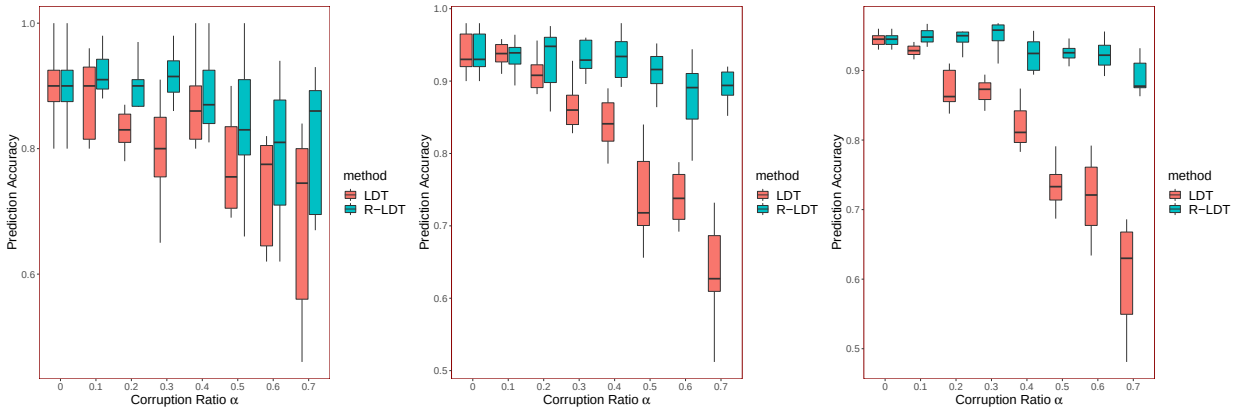
4.4.4 Results: R-LDT outperforms other robust learning strategies

To further verify the superiority of R-LDT, we compare its performance with existing robust learning methods including Huber loss based learning [280], Label noise robust SVM (LNRSVM) [281], the TORRENT (TOR) algorithm [153] and robust regression with L_q loss (L_q-RR, 282) following the line of regression models. Meanwhile, we also take two recently proposed deep learning methods, DivideMix [283] and FINE [284] into account for the comparison which are both implemented with one-dimensional ResNets [285] as backbone neural



(a) d=2, n=10 (b) d=5, n=50 (c) d=10, n=100

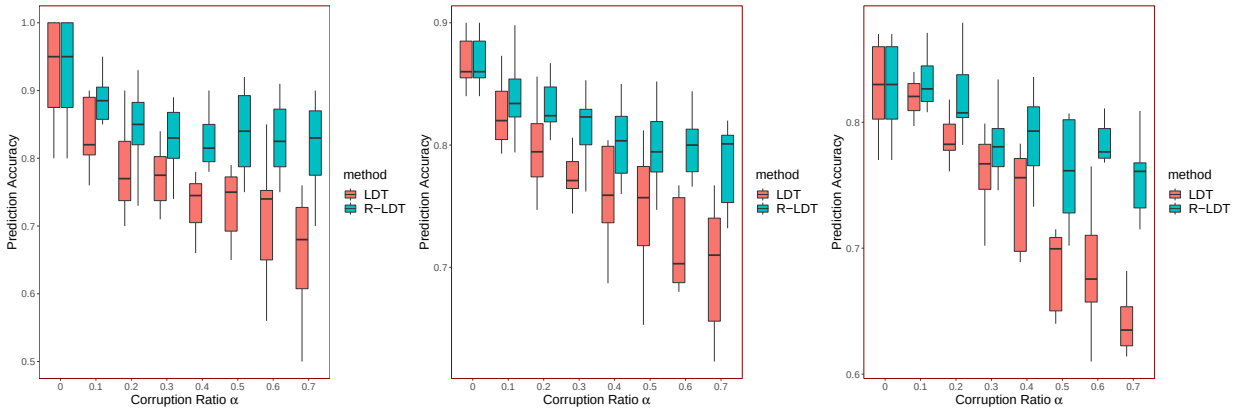
Fig. 4.5: Prediction accuracy of LDT and R-LDT on data with random rewards.



(a) d=2, n=10 (b) d=5, n=50 (c) d=10, n=100

Fig. 4.6: Prediction accuracy of LDT and R-LDT on data with contribution rewards.

networks. At last, the L_1-LDT as formulated in (4.9) is also compared with our R-LDT. In what follows we show the results in Table 4.6 on $d = 5$, $n = 50$ and $\alpha = 0.3$ and the general conclusion holds true for other parameter configuration. The MSE results of LNRSVM, DivideMix and FINE are not reported since they do not directly estimate the coefficients of the



(a) d=2, n=10 (b) d=5, n=50 (c) d=10, n=100

Fig. 4.7: Prediction accuracy of LDT and R-LDT on data with predictive rewards.

attributes. Overall, R-LDT outperforms other methods in accuracy and MSE (if any) including the state-of-the-art deep and non-deep learning methods designed for corrupted data. The only exception is that TOR can outperform R-LDT when the rewards are generated from the predictive rewards. However, it doesn't perform well when contribution rewards or random rewards are used. Its success with predictive rewards is hardly a surprise: it can be attributed to the fact that the TOR method, as a regression model, actually needs the reward as a good surrogate of the response variable to estimate the user's decision threshold, so while the predictive rewards essentially provides rewards that are close to the users' decision thresholds, this helps TOR to learn well from data. When there is no sufficient prior knowledge for using predictive rewards, the rewards set by the system might actually deviate from the users' decision thresholds, which leads to the low performance of TOR on both random rewards and contribution rewards scenarios. In contrast, R-LDT doesn't rely on the reward mechanisms, which is more resistant and enjoys wider applicability in practice. Even though its performance is worse than TOR on predictive rewards, it still outperforms other methods on this reward scenario.

One can observe that R-LDT outperforms L_1-LDT in terms of both accuracy and MSE

Table 4.6: A comparison of methods on simulated data ($d = 5, n = 50, \alpha = 0.3$).

Method	Random		Contribution		Predictive	
	Accuracy(%)	MSE	Accuracy(%)	MSE	Accuracy(%)	MSE
LR	54.12±8.37	0.359±0.010	81.92±7.46	0.540±0.096	46.09±5.87	0.601±0.039
Huber	58.49±5.76	0.261±0.105	72.20±9.56	0.617±0.121	59.92±7.32	1.149±0.102
LNRSVM	66.38±3.06	-	78.00±9.32	-	55.27±4.20	-
DivideMix	82.13±4.24	-	88.73±4.47	-	68.23±5.99	-
FINE	89.39±3.04	-	90.11±3.88	-	77.14±5.50	-
L_q-RR	80.40±4.84	0.244±0.116	81.33±4.11	0.523±0.118	70.40±4.89	0.318±0.034
LDT	87.50±3.86	0.203±0.111	86.30±4.78	0.414±0.119	78.01±3.99	0.110±0.081
L_1-LDT	89.79±3.44	0.182±0.042	89.81±4.54	0.292±0.097	79.78±3.95	0.099±0.034
TOR	83.21±6.61	0.279±0.083	58.37±8.33	1.300±0.398	**90.79±4.19**	**0.011±0.005**
R-LDT	**92.92±1.56**	**0.053±0.026**	**93.20±2.78**	**0.151±0.089**	81.82±3.27	0.051±0.024

across all cases. But it is also fair to say that empirically L_1-LDT performs fine, particularly in terms of prediction accuracy, given that we have shown in **Section 4.3.3** that there is a fundamental difficulty with L_1-LDT in our problem. It is commonly known in the context of variable selection that, L_1 norm imposes bias on the parameter estimation, while here, with the mingle of the estimations of both $\boldsymbol{\beta}$ and $\boldsymbol{b}$, the bias that L_1-LDT imposes on the estimation of $\boldsymbol{\beta}$ is even more exaggerated by the incapacity to detect the corruption points in $\boldsymbol{b}$. Recall also that there is a sensitive relationship between L_1-LDT to its regularization multiplier λ_{L_1}. Even this hyperparameter tuning is well done, the estimated parameters given by its solution can still hardly approximate the true parameters. One can confirm this point by seeing Table 4.6 that the MSE of L_1-LDT is quite larger than R-LDT, showing that indeed L_1-LDT has a hard time estimating the model parameters.

4.5 Real-world Case Study

We further compare R-LDT with LDT on a real-world dataset collected in an online experiment for the personalized Transportation Demand Management (TDM) system developed

Table 4.7: The mean prediction accuracy of LDT and R-LDT on TDM dataset.

$\boldsymbol{\alpha}$	**LDT**	**R-LDT** ($\gamma = 0.1$)	**R-LDT** ($\gamma = 0.3$)	**R-LDT** ($\gamma = 0.5$)	**R-LDT** ($\gamma = 0.7$)	**R-LDT** ($\gamma -$ CV)
0	**0.787±0.098**	0.779±0.102	0.769±0.106	0.745±0.121	0.724±0.137	0.782±0.124
0.1	0.745±0.120	**0.770±0.109**	0.767±0.112	0.746±0.131	0.729±0.140	0.768±0.121
0.3	0.674±0.074	0.694±0.072	0.701±0.081	0.717±0.092	0.706±0.108	**0.724±0.094**
0.5	0.638±0.082	0.641±0.081	0.643±0.078	0.658±0.078	**0.670±0.085**	0.665±0.072

by [254]. It consists of responses from 828 participants recruited from Amazon Mechanical Turk (AMT) platform. The participants were asked to make binary decisions in 13 scenarios, which result in 13 samples for each individual. Each sample has three attributes including scheduled delay early x_{SDE}, scheduled delay late x_{SDL} and minutes of time saving x_{TTS}, and a reward point given by the system to encourage users to make choices. As some of the users' data are corrupted, e.g., some users simply just rejected all the alternatives or exhibit no rational decision making patterns [1], these users' data could not be used for evaluation of our method due to the impossibility of obtaining a clean test dataset from these users. Thus, we use a screening algorithm called PBASE which we will show in **Section 4.6** to score the 828 users and select the top 600 users for experiments here. For the selected users, we split the data by a 7:6 train-test ratio. We then corrupt the training data following the same strategy as we used in the simulation study, i.e., $\tilde{r} = r + \delta_b$. But here, $\delta_b \sim U(100, 200)$ so that it can be compatible with the scale of this dataset. The corruption ratio α takes values from $\{0, 0.1, 0.3, 0.5\}$. Since there is no knowledge of the true model, we cannot use the mean square errors to compare models' parameter estimation performance. Thus, the only metric for evaluation in these experiments is the prediction accuracy.

Results are shown in Table 4.7. For each α, we compare the performance of LDT and R-LDT with different γ including the value selected by 5-fold cross validation (CV). On non-corrupted data, i.e., when $\alpha = 0$, LDT works well. For R-LDT, as γ is larger, the

Table 4.8: A comparison of methods on TDM dataset.

Method	**LR**	**Huber**	**LNRSVM**	**DivideMix**	**FINE**
Accuracy (%)	67.22±3.42	69.23±2.67	70.10±2.03	72.13±1.78	73.12±1.27
Method	**Lq-RR**	**LDT**	L_1**-LDT**	**TOR**	**R-LDT**
Accuracy (%)	70.86±1.85	72.35±1.93	72.86±1.16	72.31±1.99	**74.54±0.88**

mean accuracy gradually decreases, because larger γ means less data is used for training the model. When $\alpha > 0$, the data is corrupted. The performance of LDT is undermined, but R-LDT can still maintain a good performance. As we see in Table 4.7, with $\alpha = 0.1$, the mean accuracy of R-LDT can reach 0.770 with $\gamma = 0.1$, which outperforms LDT. Since α is only 0.1, assuming the existence of too many corrupted samples (for example, $\gamma = 0.7$) may weaken the algorithm instead. However, as α increases to 0.3, no matter which value γ takes, R-LDT performs better than LDT in general. And when $\gamma = 0.5$, the mean accuracy of R-LDT reaches 0.717, which is much higher than the accuracy of LDT. We also notice that when γ is chosen by cross validation, regardless of α, as shown in Table 4.7, the mean prediction accuracy for R-LDT with γ chosen by cross validation can be or close to the highest accuracy achievable for a given α. Such results show the possibility of choosing γ by cross validation on the training data to guarantee a good performance of R-LDT in practice. For the sake of page limit, we refer readers to more results in **Appendix 4.8.5**.

Similar to **Section 4.4.4**, we also compare R-LDT with other robust learning models and show the results in Table 4.8. In this experiment, the corruption ratio for each individual is randomly set varying from 0 to 0.5. Overall, our observation is consistent with the simulation studies and R-LDT outperforms other robust learning strategies.

4.6 Bad Actor Detection

In this section, we show how to detect bad actors based on R-LDT. The task of bad actor detection is to detect users whose data is severely corrupted than usual. Data from these bad

Algorithm 4.4 Potential Bad Actor Screening

Input: User data $\mathcal{D} = \{(u_i, \boldsymbol{x}_i, r_i, y_i)\}_{i=1,\cdots,\sum_{j=1}^{N} n_j}$, parameters m and γ,

Group $\mathcal{D}$ by u_i as $\mathcal{D} = \{(u^{(k)}, D_{u^{(k)}})\}_{k=1,\cdots,N}$.
$S \leftarrow \{\}, N \leftarrow |\{u_1, \cdots, u_n\}|$
for $k = 1, 2, \cdots, N$ **do**
    $\boldsymbol{\beta}_k \leftarrow$ R-LDT$(D_{u^{(k)}}, \gamma)$
    for $i = 1, 2, \cdots, n_k$ **do**
        $e_{k,i} = \min\{y_{k,i}(r_{k,i} - \boldsymbol{\beta}_k^T \boldsymbol{x}_{k,i}) - 1, 0\}$
    end for
    $l_k = [\gamma n_k]$
    $\boldsymbol{e}_k =$ TopmSort$([e_{k,1}, \cdots, e_{k,n_k}], l_k)$
    $S \leftarrow S \cup \{(k, \boldsymbol{e}_k)\}$
end for
$B =$TopmSort$(S, m,$ Score$(\cdot))$

Output: A set of possible bad users $B = \{u^{(i_1)}, \cdots, u^{(i_m)}\}$

Algorithm 4.5 TopmSort

Input: A set of data $S = \{s_i\}_{i=1,2,\cdots}$, a selection index m, a measure $f(\cdot)$

Sort S by $f(s)$ in an descending order.
Select the first m elements $s_{i_1} \succ \cdots \succ s_{i_m}$

Output: $i_1, \cdots, i_m$

actors would not help any modeling at all, because there is no regularity in their decision makings, so it is better to detect them for further scrutiny before we dive into modeling effort using their data. Although it is beyond the scope of this dissertation to discern the types of bad actors (i.e., either malicious, adversarial, or cheating), we take advantage of the generality of the learning formulation of R-LDT and show that it is straightforward to develop a system to filter the users and find a suspicious user set $B = \{u^{(i_1)}, \cdots, u^{(i_m)}\}$. This

leads to the idea of the potential bad actor screening (PBASE) approach (see Algorithm 4.4).

4.6.1 Potential bad actor screening (PBASE)

Ideally, if we know the corruption b for each sample of each user, we could filter the users based on some statistics of b. However, it is impossible to know the exact value of b in the presence of the sign function. An alternative way is to consider the one-side bound of b. As the corruption level of a sample is measured by the amount of corruption needed for reversal of the true decision label, we propose an estimate of sample corruption, i.e., $\hat{e} = \min\{y(r - \boldsymbol{\beta}^\top \boldsymbol{x}) - 1, 0\}$, which measures the effort we need to counter for the corruption of a sample. The definition of $\hat{e}$ is consistent with the bound estimate of b in Algorithm 4.3. To see that, when the observed decision label $y = 1$, $\hat{e} = \min\{r - \boldsymbol{\beta}^\top \boldsymbol{x} - y, 0\}$. And when $y = -1$, we have $\hat{e} = \min\{-(r - \boldsymbol{\beta}^\top \boldsymbol{x} - y), 0\} = -\max\{r - \boldsymbol{\beta}^\top \boldsymbol{x} - y, 0\}$.

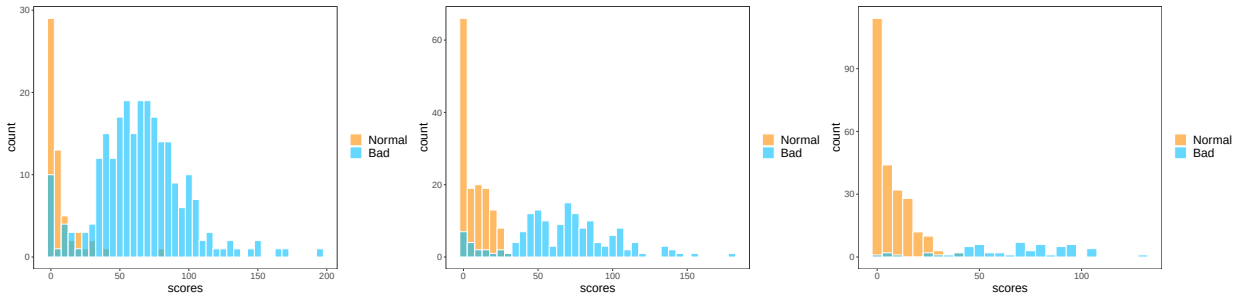
After we derive the $\hat{e}$ for all the samples of a user i, the overall corruption score of user i can be calculated using the formula:

$$Score_i = \sqrt{\frac{1}{K} \sum_{k=1}^{K} \hat{e}_{i,j_k}^2}$$

where $\{j_1, \cdots, j_K\}$ are the samples with the K largest $\hat{e}$ in terms of absolute values. Here, $K = [\gamma n]$ is the effective number of corrupted samples. Based on this score, we can screen out the users who have the greatest scores as potential bad actors by appropriately setting a threshold. There are no universal rules of the choice of this threshold as it should be carefully selected depending on the data distribution of a particular dataset, i.e., a larger threshold will probably increase the Type-II error. In practice, if there exists a validation dataset, one can choose the threshold that maximizes the difference between the True Positive Rate (TPR) and False Positive Rate (FPR), which can be visualized on the ROC curve obtained on the validation dataset. Otherwise, users need to use domain knowledge to choose a proper threshold, or can use simulation to simulate the data according to their knowledge of the process and empirically derive a reliable and effective range of the threshold, just like how the range of some parameters can be determined in the control chart literature.

Table 4.9: Mean AUC under different ρ.

ρ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
AUC	0.941	0.949	0.951	0.952	0.956	0.947	0.945	0.945	0.936



(a) $\rho = 0.2$ (b) $\rho = 0.5$ (c) $\rho = 0.8$

Fig. 4.8: Examples of corruption score histograms.

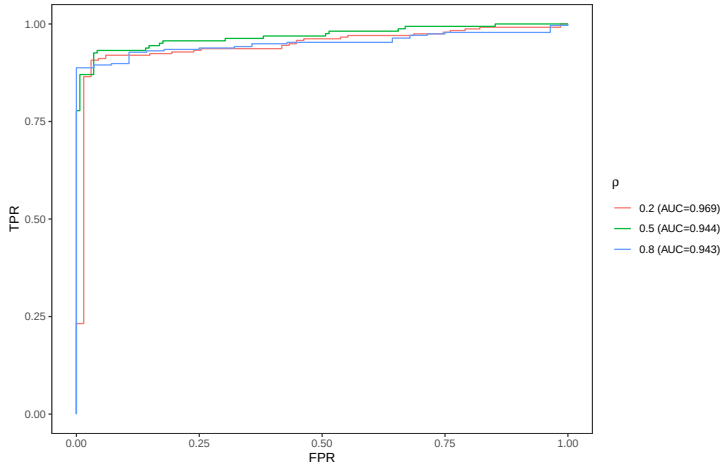


Fig. 4.9: ROC curves for bad actor detection.

4.6.2 Simulation Study

Now we show how the proposed PBASE algorithm can detect bad actors by a simulation study. Similar to **Section 4.4**, we simulate 300 users by generating their normal behavioral data. Then we create the bad actors following a Bernoulli distribution assuming the probability of any user being a bad actor is $1 - \rho$. For a bad actor i, the corruption ratio α_i is sampled from a uniform distribution, i.e., $\alpha_i \overset{i.i.d}{\sim} U(0, 0.7)$. In practice, when the bad actors and the normal users are mixed together, we can't access the true corruption ratio for each user. To distinguish bad actors from normal users, we fix the corruption parameter γ for all the users and compare their corruption scores.

To implement PBASE, we tune the parameter γ by cross validation. We run Algorithm 4.4 for 10 times and use ROC curves to evaluate the performance of PBASE. Examples of ROC curves when $\rho = 0.2, 0.5$ and 0.8 are shown in Figure 4.9. Table 4.9 reports the mean AUC scores of PBASE with various ρ. All of the scores are greater than 0.9, which suggests the proposed algorithm can accurately detect the bad actors. Figure 4.8 further shows how PBASE assigns statistically significantly different distributions of scores to bad actors and normal users. I.e., the scores of normal users are clustered on the left, which means they are predominantly small and even close to zero, while the scores of bad actors are distributed on the right indicating that they are much larger.

4.7 Conclusions

Modeling user behaviors when interacting with a reward system can help improve user-system interaction and achieve better outcomes for both the users and the system. However, the existence of data corruption in users' data or bad actors in the user population may cause difficulty in user modeling. To address this problem, based on the framework of the recently proposed Latent Decision Threshold (LDT) model, this dissertation proposes a robust learning formulation, named R-LDT, to improve the robustness of parameter learning of LDT model when the data is corrupted. We further propose a bad actor screening algorithm to identify potential bad actors in the user population. We conduct extensive numerical ex-

periments using both simulated and real-world data to demonstrate the effectiveness of our proposed methods in both parameter learning and bad actor detection comparing with a variety of other competing robust learning models.

4.8 Appendix

4.8.1 Supporting Lemmas

Lemma 4.8.1. *The loss function in (8) can be rewritten as* $L_{S_t}(\boldsymbol{\beta}) = \sum_{i\in S_t} |b_i| + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2$.

Proof. In Algorithm 4.3, if $y_i = 1$, we have

$$b_i = \min\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\} = \min\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i, 1\} - 1 \leq 0,$$

which gives $|b_i| = -\min\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\}$. Similarly, if $y_i = -1$, we have

$$b_i = \max\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\} = \max\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i, -1\} + 1 \geq 0,$$

which leads to $|b_i| = b_i = \max\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\}$. At iteration t, let $S_t = S_t^+ \cup S_t^-$, where $S_t^+ = \{i : y_i = 1\}$ and $S_t^- = \{i : y_i = -1\}$. Then, we have

$$\begin{aligned}
L_{S_t}(\boldsymbol{\beta}) &= \sum_{i\in S_t} [1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 = \sum_{i\in S_t} [y_i\{y_i - (r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)\}]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 \\
&= \sum_{i\in S_t^+} [y_i - (r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)]_+ + \sum_{i\in S_t^-} [-\{y_i - (r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)\}]_+ + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 \\
&= \sum_{i\in S_t^+} \max\{-(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i), 0\} + \sum_{i\in S_t^-} \max\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\} + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 \\
&= -\sum_{i\in S_t^+} \min\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\} + \sum_{i\in S_t^-} \max\{r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i - y_i, 0\} + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2 \\
&= \sum_{i\in S_t} |b_i| + \frac{\lambda}{2}||\boldsymbol{\beta}||_2^2
\end{aligned}$$

□

Lemma 4.8.2. *Given a partition of the observations* $A_< \cup A_\geq$, *suppose the corresponding feasible region of* $\boldsymbol{\beta}$ *is* S. *For* $\boldsymbol{\beta} \in S$, *there exists an* $\epsilon > 0$ *such that* $\boldsymbol{\beta} + \Delta_{\boldsymbol{\beta}} \in S$ *for any change* $\Delta_{\boldsymbol{\beta}}$ *on* $\boldsymbol{\beta}$ *satisfying* $||\Delta_{\boldsymbol{\beta}}||_2 \leq \epsilon$.

Proof. Let us consider $\Delta_{\boldsymbol{\beta}}$ that keeps $A_<$ unchanged. We should have

$$y_i[r_i - (\boldsymbol{\beta} + \Delta_{\boldsymbol{\beta}})^\top \boldsymbol{x}_i] < 1, \quad i \in A_<$$

which is equivalent to

$$\Delta_{\boldsymbol{\beta}}{}^\top \boldsymbol{x}_i < 1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i), \quad \forall i \in A_<$$

Since $i \in A_<$, the right-hand side should be positive. With $A_<$ being a finite set, given such a system of linear inequalities, with $\boldsymbol{\beta}$ fixed, we can easily find $z = \min_{i \in A_<}\{1 - y_i(r_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)\}$. Meanwhile, we can easily find $u = \max_{i \in A_<} ||\boldsymbol{x}_i||_2$. Obviously, $z, u > 0$. Let $\epsilon < \frac{z}{u}$. For any $||\Delta_{\boldsymbol{\beta}}||_2 < \epsilon$, with Cauchy-Schwarz inequality, we have

$$|\Delta_{\boldsymbol{\beta}}{}^\top \boldsymbol{x}_i| \leq ||\Delta_{\boldsymbol{\beta}}||_2 ||\boldsymbol{x}_i||_2 \leq \epsilon u < z$$

which gives $\Delta_{\boldsymbol{\beta}}{}^\top \boldsymbol{x}_i < z \leq$ RHS, i.e., the system of linear inequalities holds, which indicates $\boldsymbol{\beta} + \Delta_{\boldsymbol{\beta}} \in S$. Thus, we can always find a bound ϵ such that when $||\Delta_{\boldsymbol{\beta}}||_2 < \epsilon$, $\boldsymbol{\beta} + \Delta_{\boldsymbol{\beta}} \in S$, if $\boldsymbol{\beta} \in S$. □

We also include in this Appendix additional results of the experiments in **Section 4.4** and **Section 4.5**.

4.8.2 The solution path of L_1-LDT on a simulation dataset

To further compare LDT, L_1-LDT and our R-LDT, we generate a simulation dataset with $d = 5$ and $n = 50$. The corruption ratio is set to $\alpha = 0.3$. We present the solution path of L_1-LDT in Figure 4.10. It can be seen that the the five corresponding parameters cannot get close to the true values at the same time. As λ increases, some parameters gradually approximate the true value lines, while others deviate from their true values.

For comparison with LDT, we contrast the trajectories with the LDT estimations in Figure 4.11. In the figure, obviously, as λ increases, the estimations given by L_1-LDT will finally converge to the LDT estimations. This also verifies Theorem 4.3.2 in **Section 4.3.3**.

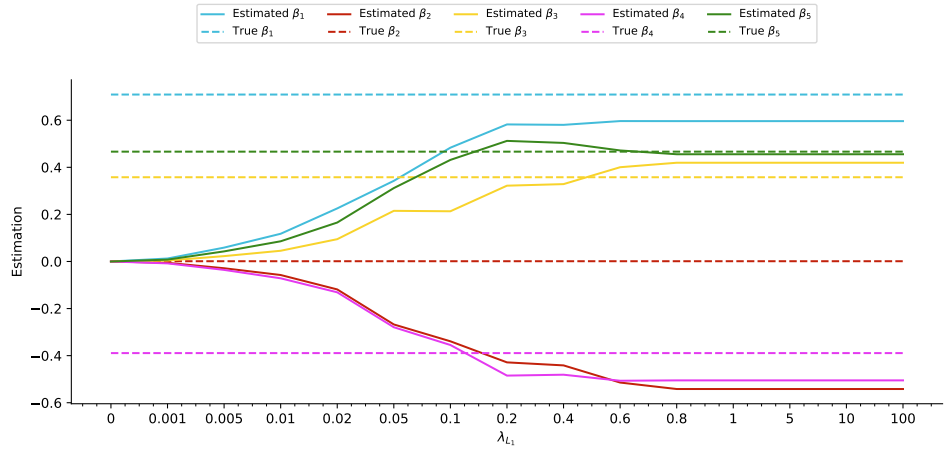


Fig. 4.10: The solution path of L_1-LDT

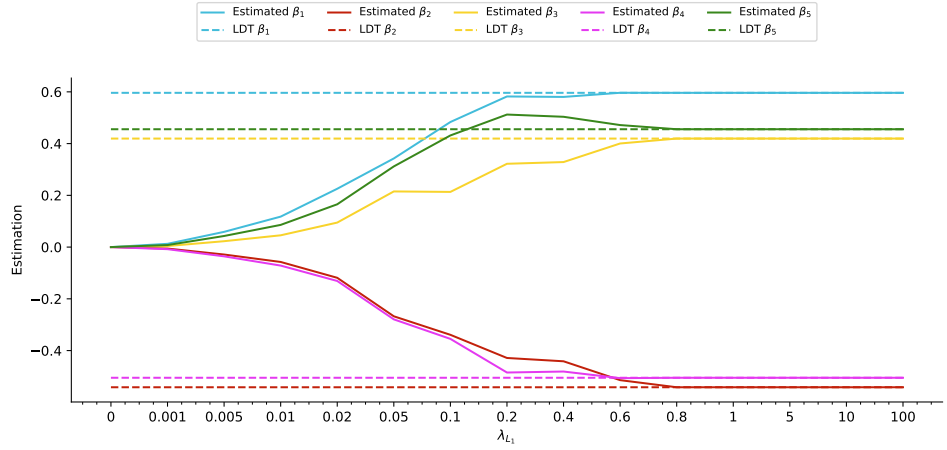


Fig. 4.11: The solution path of L_1-LDT compared with LDT.

4.8.3 More details in the experiment setup

The optimization of the LDT model learning can be addressed by any optimization solvers (e.g. Gurobi). It is worth mentioning that our formulation for LDT regularizes $\boldsymbol{\beta}$ using L_2 regularization, which also requires hyperparameter tuning. In our experiments, for simplicity, we use the same regularization hyperparameter λ by setting it to 1 for both LDT and R-LDT which seems to work well. In practice, for the models to achieve the best accuracy, tuning λ by cross validation or other model selection approaches is recommended.

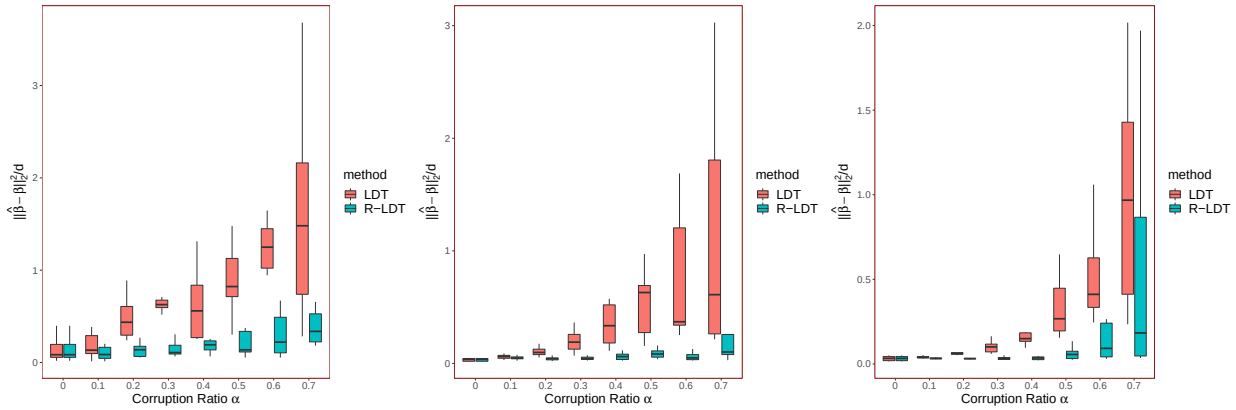
4.8.4 Additional results on parameter estimation (MSE)

Figures 4.12-4.14 present the mean square error $||\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}||_2^2/d$ of LDT and R-LDT under different settings of rewards. For both algorithms, the error shows an increasing trend as α gets larger. However, for LDT, it increases much more rapidly, but R-LDT learns more accurate model parameters than LDT. And again, there is an increasing gap between them when α increases. LDT also shows greater variability when α is large. It is of interest to see that when n is large, the magnitude of the mean square error gets smaller, but the mean square error of LDT under severe data corruption is still very large. Compared with LDT, no matter how much of the data is corrupted, the mean square error of R-LDT across the experiment settings is consistently below 0.5, which demonstrates its robustness and accuracy.

4.8.5 Detailed accuracy for Table 4.6

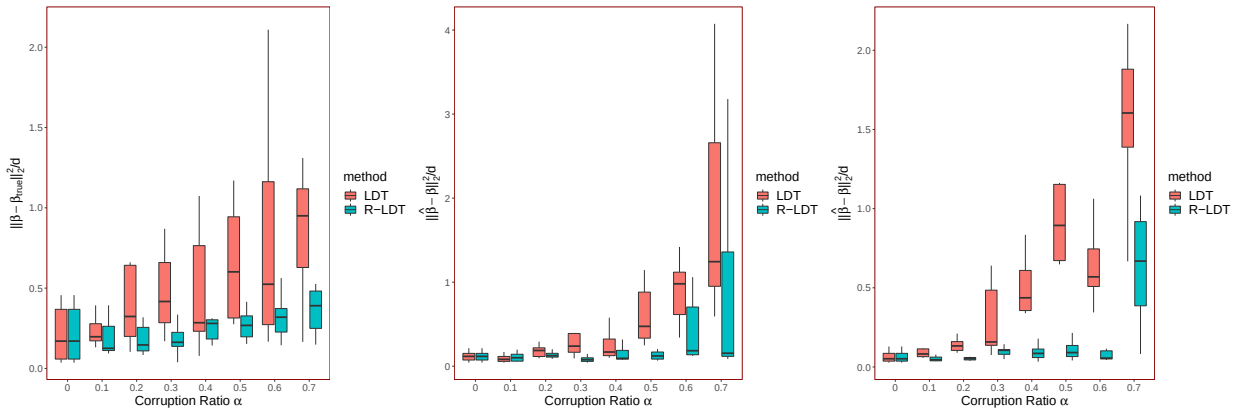
Figures 4.15-4.18 show the histograms of the prediction accuracy of LDT and R-LDT on the 600 users' data. Compared to the mean accuracy given in **Section 4.5**, these histograms provide us more details into the performance of the two methods.

When $\alpha = 0$, for most users, the prediction accuracy for both methods is greater than 0.6. And the histograms look left skewed. For R-LDT, when γ gets larger, the peak gradually decreases and there will be more bars below 0.6, which indicates the decrease of the accuracy. And the shapes of the histograms become more flat. This actually makes sense, because the data is non-corrupted, assuming there are outliers by setting larger value for γ will mislead



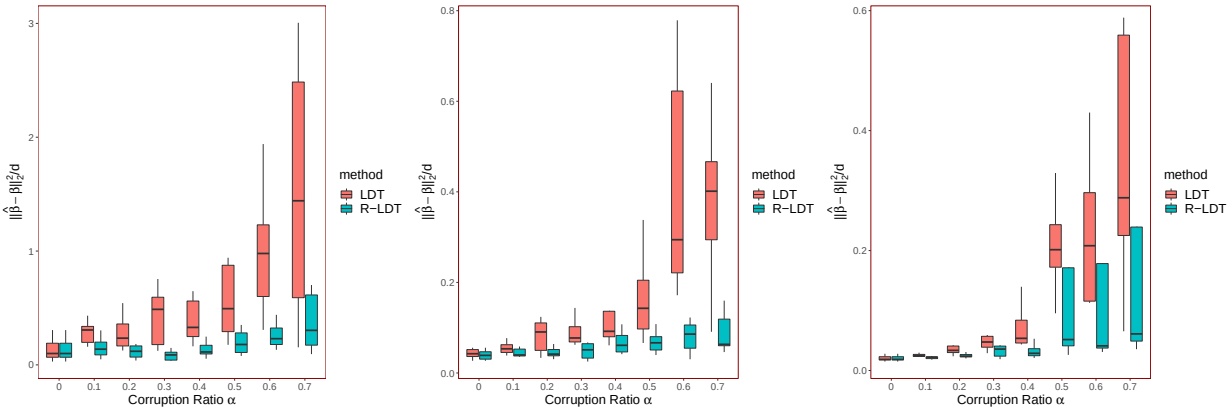
(a) d=2, n=10 (b) d=5, n=50 (c) d=10, n=100

Fig. 4.12: Comparison by $||\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}||_2^2/d$ between LDT and R-LDT on data with random rewards.



(a) d=2, n=10 (b) d=5, n=50 (c) d=10, n=100

Fig. 4.13: Comparison by $||\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}||_2^2/d$ between LDT and R-LDT on data with contribution rewards.



(a) d=2, n=10 (b) d=5, n=50 (c) d=10, n=100

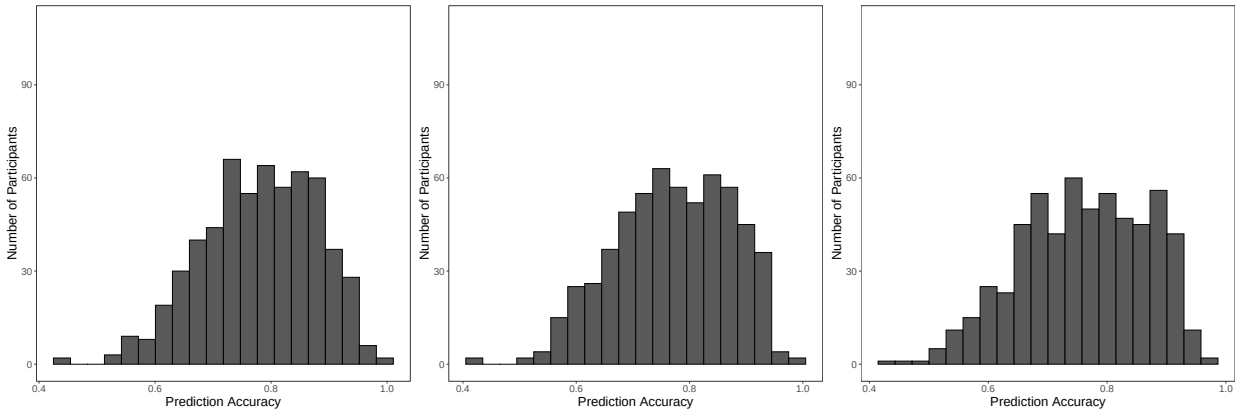
Fig. 4.14: Comparison by $||\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}||_2^2/d$ between LDT and R-LDT on data with predictive rewards.

the model to eliminate effective information and result in a drop of accuracy. On the other hand, this performance degradation is not severe and well controlled in terms of the drop of the mean accuracy as long as γ is not very large.

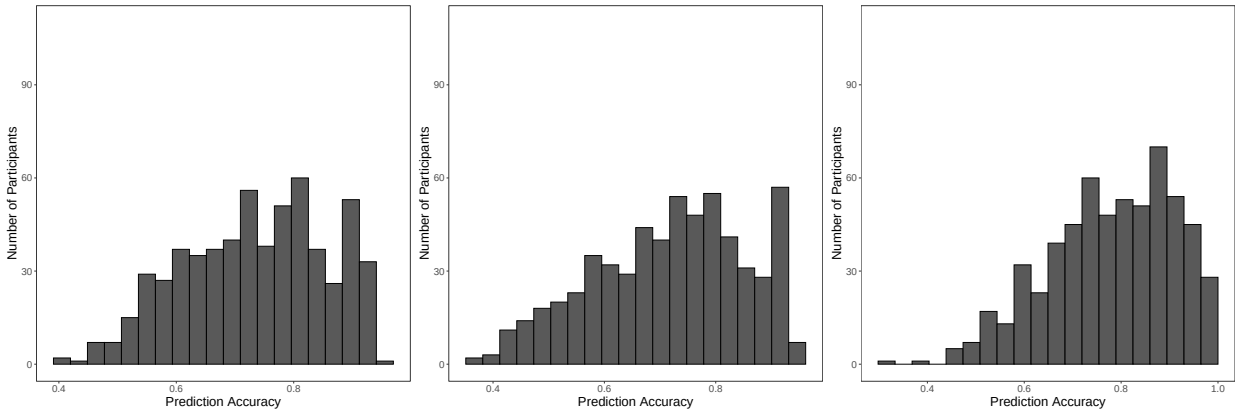
With $\alpha = 0.1$, the shape of the histogram of LDT's accuracy is concentrated around 0.75. But for R-LDT, the peak quickly shifts to the right, i.e., around 0.8, when $\gamma = 0.1$ or $\gamma = 0.3$. As γ grows , the peak is shifted to the left. It suggests that the mean performance of R-LDT should be the best when γ is small as 0.1 or 0.3.

If we increase α to 0.3, we can observe similar results. As γ gets larger, R-LDT can reach higher accuracy on more users as the peak gradually shifts to the right. But if γ is too large, e.g., reaching 0.7, which might greatly exceed the true corruption ratio, the performance might be degraded instead.

When $\alpha = 0.5$, for R-LDT, we can also see the peak shift towards right as γ increases, with more bars moving beyond 0.6. However, compared to the cases when α is smaller, the result indicates worse performance. It is due to the increase of data corruption. Larger α will lead to more data corruptions, which makes it harder to learn an accurate model.

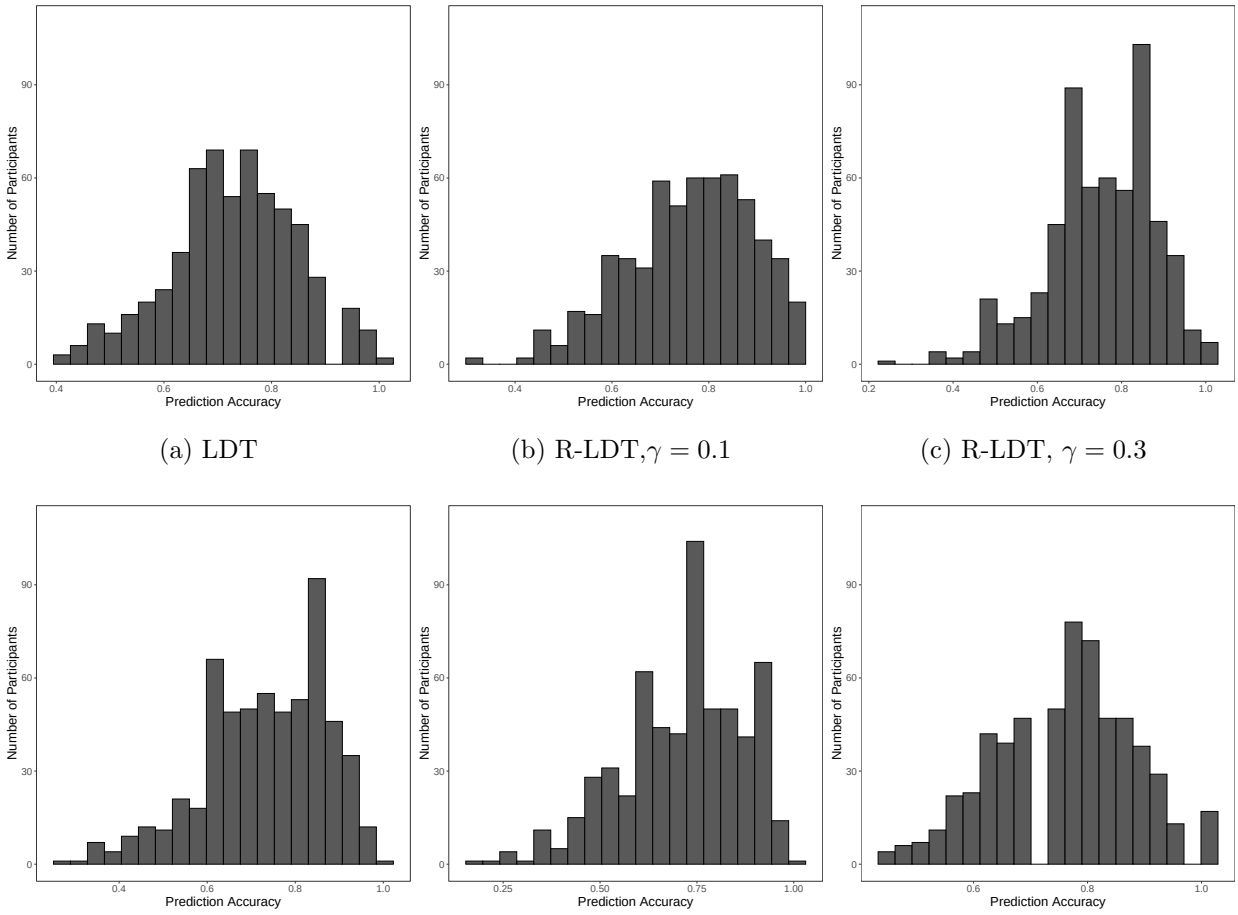


(a) LDT (b) R-LDT,$\gamma = 0.1$ (c) R-LDT, $\gamma = 0.3$



(d) R-LDT,$\gamma = 0.5$ (e) R-LDT,$\gamma = 0.7$ (f) R-LDT,$\gamma -$ CV

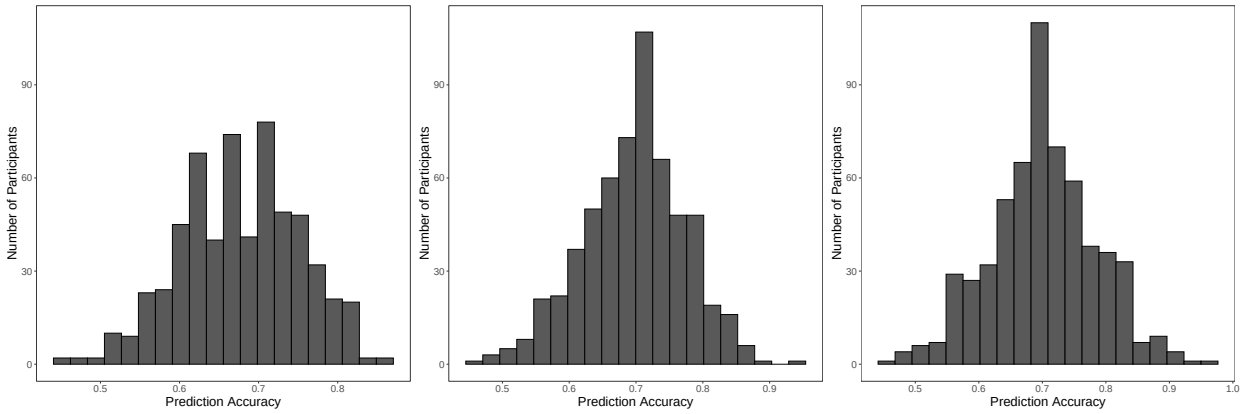
Fig. 4.15: Histograms of prediction accuracy for LDT and R-LDT ($\alpha = 0$).



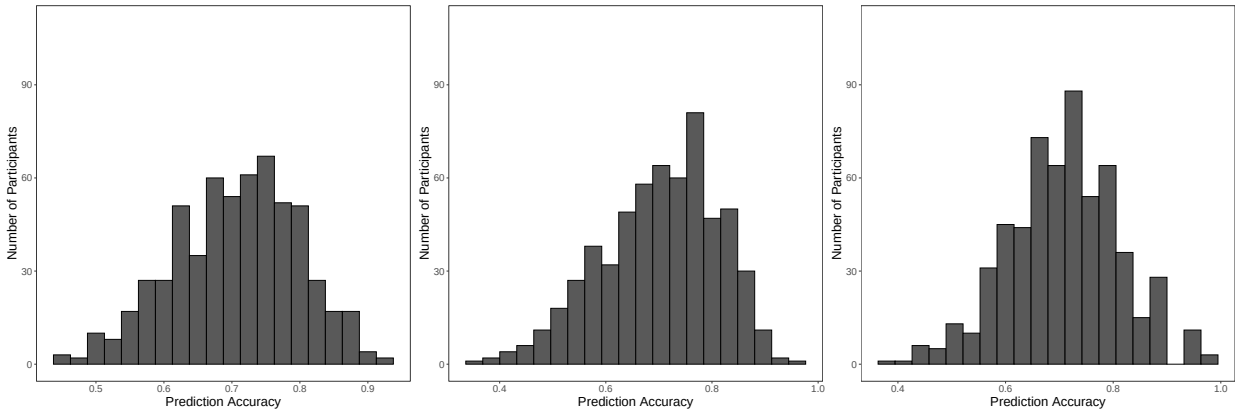
(a) LDT (b) R-LDT,$\gamma = 0.1$ (c) R-LDT, $\gamma = 0.3$

(d) R-LDT,$\gamma = 0.5$ (e) R-LDT,$\gamma = 0.7$ (f) R-LDT,$\gamma - \text{CV}$

Fig. 4.16: Histograms of prediction accuracy for LDT and R-LDT ($\alpha = 0.1$).

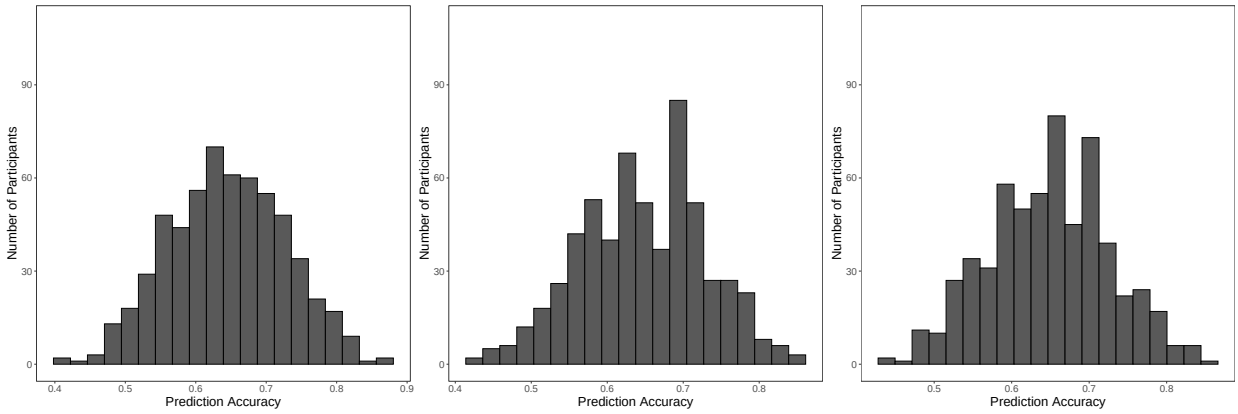


(a) LDT (b) R-LDT,$\gamma = 0.1$ (c) R-LDT, $\gamma = 0.3$

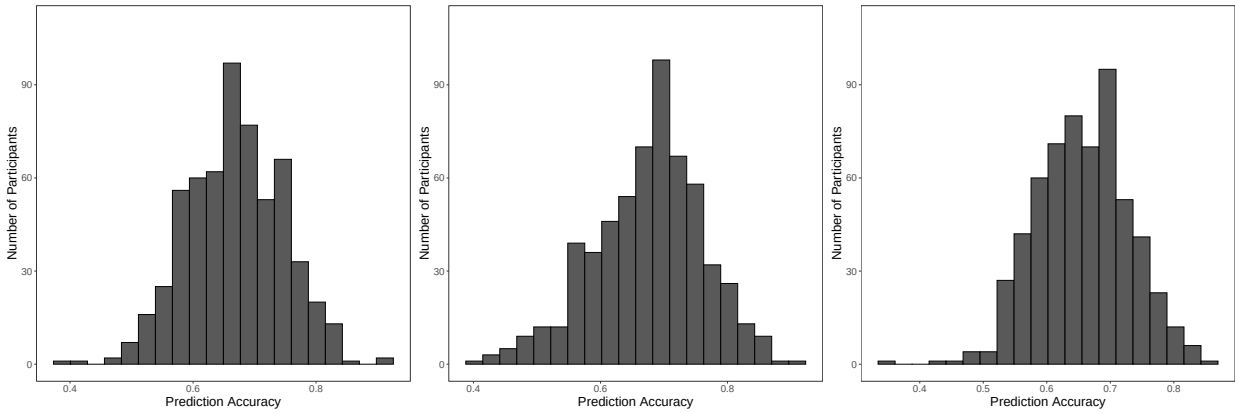


(d) R-LDT,$\gamma = 0.5$ (e) R-LDT,$\gamma = 0.7$ (f) R-LDT,$\gamma -$ CV

Fig. 4.17: Histograms of prediction accuracy for LDT and R-LDT ($\alpha = 0.3$).



(a) LDT (b) R-LDT,$\gamma = 0.1$ (c) R-LDT, $\gamma = 0.3$



(d) R-LDT,$\gamma = 0.5$ (e) R-LDT,$\gamma = 0.7$ (f) R-LDT,$\gamma -$ CV

Fig. 4.18: Histograms of prediction accuracy for LDT and R-LDT ($\alpha = 0.5$).

Chapter 5

CONCLUSION, DISCUSSION, AND FUTURE DIRECTIONS

This dissertation develops methodologies for building trustworthy personalized systems under user-system interactions. It addresses challenges arising from the three key factors central to this objective: user heterogeneity, system trustworthiness, and user-system interactions. Chapter 2 focuses on user heterogeneity and presents CrowdLLM, a framework for creating digital populations that capture the diversity and decision-making fidelity of real human populations beyond the capabilities of pure LLM-based approaches. Chapter 3 highlights the challenges of improving fairness, a key dimension of trustworthiness, in personalized ML systems, and introduces FairCL as a general framework to reduce model performance disparities for heterogeneous users. Chapter 4 further studies the robustness in reward-based user-system interactions, and presents R-LDT as a solution to the robust learning of user choice behaviors. Together, these contributions advance our understanding of how to build human-centric AI systems that can sustainably adapt to user heterogeneity while addressing key trustworthiness concerns.

5.1 Discussion

The methodologies developed in this dissertation point a way toward the development of personalized digital twins, in which each user is paired with an AI-empowered digital twin as their computational representation to support complex and lifelong decision-making in their unique contexts. In healthcare, for example, such a system could function as a personalized coach that understands an individual's health goals, anticipates health risks, and recommends reliable long-term care strategies tailored to their evolving needs, thereby supporting sustained engagement and long-term health management. Realizing this vision requires mov-

ing beyond existing one-size-fits-all ML methods that primarily optimize average predictive performance. Instead, it requires a system-level framework that can characterize population-level heterogeneity, adapt models to individual users, and remain robust in dynamic environments. This dissertation contributes to this foundation through a top-down pathway that connects population-level modeling, individual-level personalization, and system-level decision-making, and the interactions between users and systems. Specifically, digital populations first characterize heterogeneity across diverse target users, and personalized models then zoom in to adapt decision-making to individual needs and preferences. Together with robustness-aware learning and anomaly detection mechanisms, these components can support reliable decision-making in complex human-centered systems and offer methodological building blocks for realizing the broader vision of personalized digital twins.

Despite this promise, these efforts remain at an early stage. The next section discusses the limitations and outlines future research directions.

5.2 Limitations and future work.

5.2.1 Digital populations: practical implications and future directions

While Chapter 2 provides CrowdLLM as a cost-effective and scalable framework to create digital populations for crowd-based decision-making under user heterogeneity, intended to be broadly useful across a range of applications, its current realization of the framework still has limitations that point to important directions for future work. In particular, CrowdLLM should be applied with additional care and may need further adaptation in the following scenarios: *1) Highly specialized domains requiring deep expertise.* The current CrowdLLM framework does not explicitly distinguish experts from non-experts. While it is developed to capture individual-level variation, it does not model domain-specific expertise or competence, which may be critical in highly specialized settings.

In future work, CrowdLLM should be extended to better support such knowledge-intensive domains. On the one hand, the general-purpose LLM backbone for reference decision generation can be replaced by more specialized models (e.g., domain-specific foundation models,

rule-based systems, knowledge graph, cognitive models, etc.). On the other hand, the profile and belief generators can be also extended across multiple dimensions to better capture experts' diverse characteristics, including expert identification (e.g., contrastive learning, metric learning), structured modeling (e.g., mixture-of-experts, Bayesian hierarchical modeling), quality control, uncertainty-aware calibration, etc. *2) Long-horizon tasks with delayed outcomes.* CrowdLLM follows the supervised learning paradigm without considering long-term incentives. As a result, directly applying it to long-horizon strategic decision-making scenarios with delayed outcomes may be inappropriate, since optimal decisions in such settings depend on repeated interactions and future outcomes. A promising direction for extending CrowdLLM to these settings is to redesign the current belief generation module for such long-horizon decision-making scenarios. For example, reinforcement learning can be incorporated to model adaptive decision-making and learn strategic policies for tasks with long-term incentives. *3) Scenarios with longitudinal dynamic individual behaviors.* In such cases, longitudinal histories play a critical role in predicting future outcomes as they capture temporal dynamics that may not be adequately represented by cross-sectional data snapshots alone. CrowdLLM follows a static formulation without such dynamic considerations, and therefore cannot support longitudinal modeling in its current form. To address these concerns, future extensions could incorporate explicit temporal modeling mechanisms at multiple levels. The LLM backbone could be augmented with temporal reasoning mechanisms (e.g., sequence-based prompting, memory modules, etc.) to better utilize historical information. In parallel, the profile and belief generators could be extended to capture time-evolving contextual information and model the behavioral dynamics of the population (e.g., state-space formulation, diffusion models, etc.). Collectively, these scenarios do not undermine the practical value of CrowdLLM for many crowd-based decision-making applications, but rather clarify the scope of applicability of the current framework. They highlight the potential research opportunities, and provide a foundation for future work that advances creating digital populations for crowd-based decision-making as an emerging modeling paradigm beyond static simulators in the era of AI.

5.2.2 Trustworthiness in personalized ML systems and beyond

As shown in Chapter 3, FairCL has the potential to generalize across a range of applications (transportation, healthcare, and manufacturing, etc.). However, the broader success of model personalization ultimately depends on whether individuals can trust the system that learns from and serves them. Fairness is an essential part of this trust, but it is not the only requirement. Other important dimensions of trustworthiness, such as privacy, robustness to adversarial tasks, have not been fully explored in this chapter. These limitations open several directions for future work: 1) Adversarial robustness: Although FairCL can somewhat show resistance to data scarcity or noise, it remains unclear how it would perform under deliberate adversarial attacks, such as data poisoning, input manipulation, or structure-aware attacks [286, 287, 288]. Personalization frameworks such as FairCL typically allow information to be shared across users to varying degrees and at different levels. As a result, attacks targeting a small number of users may affect not only their own personalized models, but also the models of other users through the sharing structure. Therefore, future work should investigate defense strategies against specific types of attacks and their potential propagation. 2) Privacy: As there has been growing concerns on data security and privacy in machine learning, it is worth exploring how to enable privacy preserving information sharing in FairCL or other model personalization frameworks. The information sharing structure such as canonical models in FairCL may involve the exchange or aggregation of information derived from sensitive user data. This concern is especially important in applications where individual data may contain private information such as demographics, mobility patterns, health records, or other sensitive attributes. As a result, a promising direction is to improve the applicability of such frameworks to privacy-sensitive settings. One possible way to do so is combining it with federated learning, which awaits more investigation in the future. 3) Scalability: When the size of the user population scales up to tens of thousands or more, more efficient algorithms are needed to improve the scalability for personalization. Many real-world systems in need of personalization operate at large scale and therefore demand

frequent model updates and efficient information exchange, and substantial uncertainty. As new users and new observations arrive continuously, maintaining and updating personalized models can become computationally expensive. Therefore, scalable personalization will be a key direction for future research. 4) Other challenges: Future work should also consider additional dimensions of trustworthiness in personalized modeling, including interpretability, safety, and distributional robustness. For long-term deployment, careful evaluation of user acceptance, feedback quality, and potential risk is essential, especially when personalized decisions may affect future user behavior. Addressing these challenges will be important for building personalized ML systems with meaningful real-world impact.

5.2.3 Modeling of user-system interactions in reward systems

Chapter 4 studies the robust learning of user choice behaviors when interaction data are potentially corrupted under a simplified offline setting. However, in many real-world reward systems, as users receive offers, respond to incentives, and learn how the system operates, their behaviors can evolve continuously. Teasing behaviors of bad actors that lead to data corruption may also emerge gradually rather than appearing as a static pattern. For future research, it is natural to extend the framework of R-LDT to an online setting for dynamic detection of suspicious behaviors. This extension would enable the system to actively monitor changes in user responses, identify potential bad actors earlier, and adapt its decision-making policy before the bad actors substantially affect the operation.

While bad actor detection supported by R-LDT is useful for protecting the system from potential data corruption, detection alone is insufficient, because problematic behaviors (e.g., teasing or inconsistent responses) may arise for different reasons. Some users may intentionally manipulate the system for extra rewards, whereas others may not be malicious but may misunderstand the task, have low engagement, or fail to recognize how their actions affect long-term outcomes. This distinction is important, because treating all such users as bad actors may reduce malicious behaviors but also harm benign user, which will weaken long-term performance and user trust.

In the future, we need to move beyond detection toward more nuanced intervention and user guidance. For malicious users, the system can eliminate them and reduce strategic manipulation through stricter monitoring and incentive redesign. For low-achieving or inconsistent users, however, a more constructive response is to provide feedback, explanations, or coaching that helps them understand the system better and interact with it more effectively. One promising direction is to incorporate reinforcement learning or other sequential decision-making approaches to enable a more active formulation for customizing incentives, feedback, or explanations for different user types. This would shift the system from passively modeling user choices to actively supporting healthier user-system interactions. Ultimately, it could facilitate long-term cooperation between users and systems and help achieve a sustainable win-win outcome.

BIBLIOGRAPHY

[1] Jingshuo Feng, Shuai Huang, and Cynthia Chen. Modeling user interaction with app-based reward system: A graphical model approach integrated with max-margin learning. *Transportation Research Part C: Emerging Technologies*, 120:102814, 2020.

[2] Shanhe Lou, Zhongxu Hu, Yiran Zhang, Yixiong Feng, Mengchu Zhou, and Chen Lv. Human-cyber-physical system for industry 5.0: A review from a human-centric perspective. *IEEE Trans. Autom. Sci. Eng*, pages 1–18, 2024.

[3] Advait Deshpande and Helen Sharp. Responsible ai systems: who are the stakeholders? In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, pages 227–236, 2022.

[4] Ozlem Ozmen Garibay, Brent Winslow, Salvatore Andolina, Margherita Antona, Anja Bodenschatz, Constantinos Coursaris, Gregory Falco, Stephen M Fiore, Ivan Garibay, Keri Grieman, et al. Six human-centered artificial intelligence grand challenges. *International Journal of Human–Computer Interaction*, 39(3):391–437, 2023.

[5] Wei Xu, Marvin J Dainoff, Liezhong Ge, and Zaifeng Gao. Transitioning to human interaction with ai systems: New challenges and opportunities for hci professionals to enable human-centered ai. *International Journal of Human–Computer Interaction*, 39(3):494–518, 2023.

[6] Lu Cheng, Kush R Varshney, and Huan Liu. Socially responsible ai algorithms: Issues, purposes, and challenges. *Journal of Artificial Intelligence Research*, 71:1137–1181, 2021.

[7] Nikolaos Polatidis and Christos K Georgiadis. Recommender systems: The importance of personalization in e-business environments. *International Journal of E-Entrepreneurship and Innovation (IJEEI)*, 4(4):32–46, 2013.

[8] Jaime Teevan, Susan T Dumais, and Eric Horvitz. Potential for personalization. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 17(1):1–31, 2010.

[9] Julian McAuley. *Personalized Machine Learning*. Cambridge University Press, 2022.

[10] Mikhail Bilenko and Matthew Richardson. Predictive client-side profiles for personalized advertising. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 413–421, 2011.

[11] Liyi Guo, Junqi Jin, Haoqi Zhang, Zhenzhe Zheng, Zhiye Yang, Zhizhuang Xing, Fei Pan, Lvyin Niu, Fan Wu, Haiyang Xu, et al. We know what you want: An advertising strategy recommender system for online advertising. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2919–2927, 2021.

[12] Javier Andreu-Perez, Daniel R Leff, Henry MD Ip, and Guang-Zhong Yang. From wearable sensors to smart implants—toward pervasive and personalized healthcare. *IEEE TBME*, 62(12):2750–2762, 2015.

[13] Noura S Abul-Husn and Eimear E Kenny. Personalized medicine and the power of electronic health records. *Cell*, 177(1):58–69, 2019.

[14] Daiqi Gao, Yufeng Liu, and Donglin Zeng. Non-asymptotic properties of individualized treatment rules from sequentially rule-adaptive trials. *JMLR*, 23(250):1–42, 2022.

[15] Zhengyi Ma, Zhicheng Dou, Yutao Zhu, Hanxun Zhong, and Ji-Rong Wen. One chatbot per person: Creating personalized chatbots based on implicit user profiles. In *SIGIR*, 2021.

[16] Mauajama Firdaus, Arunav Shandilya, Asif Ekbal, and Pushpak Bhattacharyya. Being polite: Modeling politeness variation in a personalized dialog agent. *IEEE TCSS*, 2022.

[17] Kayode Sakariyah Adewole, Nor Badrul Anuar, Amirrudin Kamsin, Kasturi Dewi Varathan, and Syed Abdul Razak. Malicious accounts: Dark of the social networks. *Journal of Network and Computer Applications*, 79:41–67, 2017.

[18] A Anju, K Shalini, Haritha Ravikumar, P Saranya, M Krishnamurthy, et al. Detection of insider threats using deep learning. In *2023 3rd International Conference on Pervasive Computing and Social Networking (ICPCSN)*, pages 264–269. IEEE, 2023.

[19] Yuxuan Lu, Jing Huang, Yan Han, Bennet Bei, Yaochen Xie, Dakuo Wang, Jessie Wang, and Qi He. Beyond believability: Accurate human behavior simulation with fine-tuned LLMs. *arXiv preprint arXiv:2503.20749*, 2025.

[20] Seth Karten, Wenzhe Li, Zihan Ding, Samuel Kleiner, Yu Bai, and Chi Jin. LLM economist: Large population models and mechanism design in multi-agent generative simulacra. In *NeurIPS 2025 Workshop on Algorithmic Collective Action*, 2025.

[21] Eva Eigner and Thorsten Händler. Determinants of LLM-assisted decision-making. *arXiv preprint arXiv:2402.17385*, 2024.

[22] Yan Leng and Yuan Yuan. Do LLM agents exhibit social behavior? *arXiv preprint arXiv:2312.15198*, 2023.

[23] Ngoc Bui, Hieu Trung Nguyen, Shantanu Kumar, Julian Theodore, Weikang Qiu, Viet Anh Nguyen, and Rex Ying. Mixture-of-personas language models for population simulation. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

[24] Chengxing Xie, Canyu Chen, Feiran Jia, Ziyu Ye, Shiyang Lai, Kai Shu, Jindong Gu, Adel Bibi, Ziniu Hu, David Jurgens, et al. Can large language model agents simulate

human trust behavior? *Advances in neural information processing systems*, 37:15674–15729, 2024.

[25] Jinghua Piao, Yuwei Yan, Jun Zhang, Nian Li, Junbo Yan, Xiaochong Lan, Zhihong Lu, Zhiheng Zheng, Jing Yi Wang, Di Zhou, et al. Agentsociety: Large-scale simulation of LLM-driven generative agents advances understanding of human behaviors and society. *arXiv preprint arXiv:2502.08691*, 2025.

[26] Xinnong Zhang, Jiayu Lin, Xinyi Mou, Shiyue Yang, Xiawei Liu, Libo Sun, Hanjia Lyu, Yihang Yang, Weihong Qi, Yue Chen, et al. Socioverse: A world model for social simulation powered by LLM agents and a pool of 10 million real-world users. *arXiv preprint arXiv:2504.10157*, 2025.

[27] Marcel Binz, Elif Akata, Matthias Bethge, Franziska Brändle, Fred Callaway, Julian Coda-Forno, Peter Dayan, Can Demircan, Maria K Eckstein, Noémi Éltető, et al. Centaur: a foundation model of human cognition. *arXiv preprint arXiv:2410.20268*, 2024.

[28] Yuan Gao, Dokyun Lee, Gordon Burtch, and Sina Fazelpour. Take caution in using LLMs as human surrogates. *Proceedings of the National Academy of Sciences*, 122(24):e2501660122, 2025.

[29] Vishakh Padmakumar and He He. Does writing with language models reduce content diversity? In *International Conference on Learning Representations*, 2023.

[30] Jiaju Chen, Chongming Gao, Shuai Yuan, Shuchang Liu, Qingpeng Cai, and Peng Jiang. Dlcrec: A novel approach for managing diversity in LLM-based recommender systems. In *Proceedings of the 18th ACM International Conference on Web Search and Data Mining*, pages 857–865, 2025.

[31] Yiming Zhang, Harshita Diddee, Susan Holm, Hanchen Liu, Xinyue Liu, Vinay Samuel,

Barry Wang, and Daphne Ippolito. Noveltybench: Evaluating creativity and diversity in language models. *arXiv preprint arXiv:2504.05228*, 2025.

[32] Yijiang River Dong, Tiancheng Hu, and Nigel Collier. Can LLM be a personalized judge? *arXiv preprint arXiv:2406.11657*, 2024.

[33] Qihan Wang, Shidong Pan, Tal Linzen, and Emily Black. Multilingual prompting for improving LLM generation diversity. *arXiv preprint arXiv:2505.15229*, 2025.

[34] Alexander Shypula, Shuo Li, Botong Zhang, Vishakh Padmakumar, Kayo Yin, and Osbert Bastani. Evaluating the diversity and quality of LLM generated content. *arXiv preprint arXiv:2504.12522*, 2025.

[35] Zhaoming Liu. Cultural bias in large language models: A comprehensive analysis and mitigation strategies. *Journal of Transcultural Communication*, 3(2):224–244, 2025.

[36] Long Mai and Julie Carson-Berndsen. Improving linguistic diversity of large language models with possibility exploration fine-tuning. *arXiv preprint arXiv:2412.03343*, 2024.

[37] Jeff Howe et al. The rise of crowdsourcing. *Wired magazine*, 14(6):176–183, 2006.

[38] Luigia Costabile, Gian Marco Orlando, Valerio La Gatta, and Vincenzo Moscato. Assessing the potential of generative agents in crowdsourced fact-checking. *arXiv preprint arXiv:2504.19940*, 2025.

[39] Daniil Moskovskiy, Sergey Pletenev, and Alexander Panchenko. LLMs to replace crowdsourcing for parallel data creation? the case of text detoxification. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14361–14373, 2024.

[40] Veniamin Veselovsky, Manoel Horta Ribeiro, Philip J Cozzolino, Andrew Gordon, David Rothschild, and Robert West. Prevalence and prevention of large language model use in crowd work. *Communications of the ACM*, 68(3):42–47, 2025.

[41] Tongshuang Wu, Haiyi Zhu, Maya Albayrak, Alexis Axon, Amanda Bertsch, Wenxing Deng, Ziqi Ding, Bill Guo, Sireesh Gururaja, Tzu-Sheng Kuo, et al. LLMs as workers in human-computational algorithms? replicating crowdsourcing pipelines with LLMs. *arXiv preprint arXiv:2307.10168*, 2023.

[42] Xia Zeng, David La Barbera, Kevin Roitero, Arkaitz Zubiaga, and Stefano Mizzaro. Combining large language models and crowdsourcing for hybrid human-ai misinformation detection. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2332–2336, 2024.

[43] Jiyi Li. Human-LLM hybrid text answer aggregation for crowd annotations. *arXiv preprint arXiv:2410.17099*, 2024.

[44] Jiyi Li. A comparative study on annotation quality of crowdsourcing and LLM via label aggregation. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6525–6529, 2024.

[45] Takumi Tamura, Hiroyoshi Ito, Satoshi Oyama, and Atsuyuki Morishima. Simulation-based exploration for aggregation algorithms in human+ ai crowd: What factors should we consider for better results? In *AAAI HCOMP*, 2024.

[46] An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. On generative agents in recommendation. In *Proceedings of the 47th international ACM SIGIR conference on research and development in Information Retrieval*, pages 1807–1817, 2024.

[47] Lei Wang, Jingsen Zhang, Hao Yang, Zhi-Yuan Chen, Jiakai Tang, Zeyu Zhang, Xu Chen, Yankai Lin, Hao Sun, Ruihua Song, et al. User behavior simulation with large language model-based agents. *ACM Transactions on Information Systems*, 43(2):1–37, 2025.

[48] Nathan Corecco, Giorgio Piatti, Luca A Lanzendörfer, Flint Xiaofeng Fan, and Roger Wattenhofer. Suber: An rl environment with simulated human behavior for recommender systems. *arXiv preprint arXiv:2406.01631*, 2024.

[49] Zijian Zhang, Shuchang Liu, Ziru Liu, Rui Zhong, Qingpeng Cai, Xiangyu Zhao, Chunxu Zhang, Qidong Liu, and Peng Jiang. LLM-powered user simulator for recommender system. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 13339–13347, 2025.

[50] Sarah H Cen, Andrew Ilyas, Hedi Driss, Charlotte Park, Aspen Hopkins, Chara Podimata, et al. Large-scale, longitudinal study of large language models during the 2024 us election season. *arXiv preprint arXiv:2509.18446*, 2025.

[51] Leah von der Heyde, Anna-Carolina Haensch, Alexander Wenz, and Bolei Ma. United in diversity? contextual biases in LLM-based predictions of the 2024 european parliament elections. *arXiv preprint arXiv:2409.09045*, 2024.

[52] Srijoni Majumdar, Edith Elkind, and Evangelos Pournaras. Generative ai voting: fair collective choice is resilient to LLM biases and inconsistencies. *arXiv preprint arXiv:2406.11871*, 2024.

[53] Sarah Ball, Simeon Allmendinger, Frauke Kreuter, and Niklas Kühl. Human preferences in large language model latent space: A technical analysis on the reliability of synthetic data in voting outcome prediction. *arXiv preprint arXiv:2502.16280*, 2025.

[54] Joshua C Yang, Damian Dalisan, Marcin Korecki, Carina I Hausladen, and Dirk Helbing. LLM voting: Human choices and ai collective decision-making. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, volume 7, pages 1696–1708, 2024.

[55] Xiutian Zhao, Ke Wang, and Wei Peng. An electoral approach to diversify LLM-based multi-agent collective decision-making. *arXiv preprint arXiv:2410.15168*, 2024.

[56] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[57] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27, 2014.

[58] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[59] Zezeng Li, Shenghao Li, Zhanpeng Wang, Na Lei, Zhongxuan Luo, and David Xianfeng Gu. Dpm-ot: a new diffusion probabilistic model based on optimal transport. In *Proceedings of the ieee/cvf international conference on computer vision*, pages 22624–22633, 2023.

[60] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

[61] Tao Zhang, Tianqing Zhu, Jing Li, Mengde Han, Wanlei Zhou, and S Yu Philip. Fairness in semi-supervised learning: Unlabeled data help to reduce discrimination. *IEEE TKDE*, 34(4):1763–1774, 2020.

[62] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *CSUR*, 54(6):1–35, 2021.

[63] Sam Corbett-Davies and Sharad Goel. The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv preprint arXiv:1808.00023*, 2018.

[64] Simon Caton and Christian Haas. Fairness in machine learning: A survey. *arXiv preprint arXiv:2010.04053*, 2020.

[65] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *International Conference on Machine Learning*, pages 325–333. PMLR, 2013.

[66] Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 797–806, 2017.

[67] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268, 2015.

[68] Zhimeng Jiang, Xiaotian Han, Chao Fan, Fan Yang, Ali Mostafavi, and Xia Hu. Generalized demographic parity for group fairness. In *International Conference on Learning Representations*, 2022.

[69] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in Neural Information Processing Systems*, 2016.

[70] Richard Berk, Hoda Heidari, Shahin Jabbari, Michael Kearns, and Aaron Roth. Fairness in criminal justice risk assessments: The state of the art. *SMR*, 50(1):3–44, 2021.

[71] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *ICTS*, 2012.

[72] Will Fleisher. What's fair about individual fairness? In *AIES*, 2021.

[73] Daniel Alabi, Nicole Immorlica, and Adam Kalai. Unleashing linear optimizers for group-fair learning and optimization. In *COLT*, 2018.

[74] Andrew Cotter, Heinrich Jiang, Maya R Gupta, Serena Wang, Taman Narayan, Seungil You, and Karthik Sridharan. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *JMLR*, 20(172):1–59, 2019.

[75] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, and Krishna P Gummadi. Fairness constraints: A flexible approach for fair classification. *JMLR*, 20(1):2737–2778, 2019.

[76] Evgenii Chzhen, Christophe Denis, Mohamed Hebiri, Luca Oneto, and Massimiliano Pontil. Fair regression with wasserstein barycenters. *Advances in Neural Information Processing Systems*, 2020.

[77] Chen Zhao and Feng Chen. Rank-based multi-task learning for fair regression. In *ICDM*, 2019.

[78] Robert Cudeck. Mixed-effects models in the study of individual differences with repeated measures data. *MBR*, 31(3):371–403, 1996.

[79] Nan M Laird and James H Ware. Random-effects models for longitudinal data. *Biometrics*, pages 963–974, 1982.

[80] Qiang Yang, Yu Zhang, Wenyuan Dai, and Sinno Jialin Pan. *Transfer learning*. Cambridge University Press, 2020.

[81] Min Jiang, Zhenzhong Wang, Shihui Guo, Xing Gao, and Kay Chen Tan. Individual-based transfer learning for dynamic multiobjective optimization. *IEEE TCYB*, 51(10):4968–4981, 2020.

[82] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE transactions on knowledge and data engineering*, 34(12):5586–5609, 2021.

[83] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust

federated learning through personalization. In *International conference on machine learning*, pages 6357–6368. PMLR, 2021.

[84] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

[85] Ying Lin, Kaibo Liu, Eunshin Byon, Xiaoning Qian, and Shuai Huang. Domain-knowledge driven cognitive degradation modeling for alzheimer's disease. In *SDM*, 2015.

[86] Jingshuo Feng, Xi Zhu, Feilong Wang, Shuai Huang, and Cynthia Chen. A learning framework for personalized random utility maximization (rum) modeling of user behavior. *IEEE TASE*, 19(1):510–521, 2020.

[87] Ying Lin, Kaibo Liu, Eunshin Byon, Xiaoning Qian, Shan Liu, and Shuai Huang. A collaborative learning framework for estimating many individualized regression models in a heterogeneous population. *IEEE TRel*, 67(1):328–341, 2017.

[88] Xu Miao, Chun-Te Chu, Lijun Tang, Yitong Zhou, Joel Young, and Anmol Bhasin. Distributed personalization. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1989–1998, 2015.

[89] Joost Verbraeken, Matthijs Wolting, Jonathan Katzy, Jeroen Kloppenburg, Tim Verbelen, and Jan S Rellermeyer. A survey on distributed machine learning. *Acm computing surveys (csur)*, 53(2):1–33, 2020.

[90] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[91] Ji Liu, Jizhou Huang, Yang Zhou, Xuhong Li, Shilei Ji, Haoyi Xiong, and Dejing Dou. From distributed machine learning to federated learning: A survey. *Knowledge and Information Systems*, 64(4):885–917, 2022.

[92] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[93] Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. Survey of personalization techniques for federated learning. In *Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 794–797, 2020.

[94] Xu Zhang, Yinchuan Li, Wenpeng Li, Kaiyang Guo, and Yunfeng Shao. Personalized federated learning via variational bayesian inference. In *International Conference on Machine Learning*, pages 26293–26310. PMLR, 2022.

[95] Michael Kamp, Jonas Fischer, and Jilles Vreeken. Federated learning from small datasets. In *International Conference on Learning Representations*, 2023.

[96] Ohad Shamir and Nathan Srebro. Distributed stochastic optimization and learning. In *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 850–857. IEEE, 2014.

[97] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE transactions on neural networks and learning systems*, 32(8):3710–3722, 2020.

[98] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[99] Avishek Ghosh, Justin Hong, Dong Yin, and Kannan Ramchandran. Robust federated learning in a heterogeneous environment. *arXiv preprint arXiv:1906.06629*, 2019.

[100] Manan Mehta and Chenhui Shao. A greedy agglomerative framework for clustered federated learning. *IEEE Transactions on Industrial Informatics*, 2023.

[101] Christopher Briggs, Zhong Fan, and Peter Andras. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, 2020.

[102] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33:19586–19597, 2020.

[103] Zaobo He, Lintao Wang, and Zhipeng Cai. Clustered federated learning with adaptive local differential privacy on heterogeneous iot data. *IEEE Internet of Things Journal*, 2023.

[104] Mahdi Morafah, Saeed Vahidian, Weijia Wang, and Bill Lin. Flis: Clustered federated learning via inference similarity for non-iid data distribution. *IEEE Open Journal of the Computer Society*, 4:109–120, 2023.

[105] Yichen Ruan and Carlee Joe-Wong. Fedsoft: Soft clustered federated learning with proximal local updating. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 8124–8131, 2022.

[106] Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34:15434–15447, 2021.

[107] Soulef Benhamdi, Abdesselam Babouri, and Raja Chiky. Personalized recommender system for e-learning environment. *Education and Information Technologies*, 22(4):1455–1477, 2017.

[108] Srikar Amara and R Raja Subramanian. Collaborating personalized recommender system and content-based recommender system using textcorpus. In *ICACCS*, 2020.

[109] Benjamin M Knisely, Monifa Vaughn-Cooke, Lee-Ann Wagner, and Jeffrey C Fink. Device personalization for heterogeneous populations: leveraging physician expertise and national population data to identify medical device patient user groups. *UMUAI*, 31(5):979–1025, 2021.

[110] Frank Emmert-Streib and Matthias Dehmer. A machine learning perspective on personalized medicine: an automized, comprehensive knowledge base with ontology for pattern recognition. *Machine Learning and Knowledge Extraction*, 1(1):149–156, 2018.

[111] Ganjar Alfian, Muhammad Syafrudin, Muhammad Fazal Ijaz, M Alex Syaekhoni, Norma Latif Fitriyani, and Jongtae Rhee. A personalized healthcare monitoring system for diabetic patients by utilizing ble-based sensors and real-time data processing. *Sensors*, 18(7):2183, 2018.

[112] Gah-Yi Ban and N Bora Keskin. Personalized dynamic pricing with machine learning: High-dimensional features and heterogeneous elasticity. *Management Science*, 67(9):5549–5568, 2021.

[113] Hema Yoganarasimhan. Search personalization using machine learning. *Management Science*, 66(3):1045–1070, 2020.

[114] Qi Liu, Shiwei Tong, Chuanren Liu, Hongke Zhao, Enhong Chen, Haiping Ma, and Shijin Wang. Exploiting cognitive structure for adaptive learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 627–635, 2019.

[115] Gongwen Xu, Guangyu Jia, Lin Shi, and Zhijun Zhang. Personalized course recommendation system fusing with knowledge graph and collaborative filtering. *Computational Intelligence and Neuroscience*, 2021, 2021.

[116] Maxim Naumov, Dheevatsa Mudigere, Hao-Jun Michael Shi, Jianyu Huang, Narayanan Sundaraman, Jongsoo Park, Xiaodong Wang, Udit Gupta, Carole-Jean Wu, Alisson G

Azzolini, et al. Deep learning recommendation model for personalization and recommendation systems. *arXiv preprint arXiv:1906.00091*, 2019.

[117] Igbe Tobore, Jingzhen Li, Liu Yuhang, Yousef Al-Handarish, Abhishek Kandwal, Zedong Nie, Lei Wang, et al. Deep learning intervention for health care challenges: some biomedical domain considerations. *JMIR mHealth and uHealth*, 7(8):e11966, 2019.

[118] Yuxin Shi, Han Yu, and Cyril Leung. Towards fairness-aware federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[119] Yuxin Shi, Zelei Liu, Zhuan Shi, and Han Yu. Fairness-aware client selection for federated learning. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 324–329. IEEE, 2023.

[120] Tiansheng Huang, Weiwei Lin, Wentai Wu, Ligang He, Keqin Li, and Albert Y Zomaya. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Transactions on Parallel and Distributed Systems*, 32(7):1552–1564, 2020.

[121] Miao Yang, Ximin Wang, Hongbin Zhu, Haifeng Wang, and Hua Qian. Federated learning with class imbalance reduction. In *2021 29th European Signal Processing Conference (EUSIPCO)*, pages 2174–2178. IEEE, 2021.

[122] Tiansheng Huang, Weiwei Lin, Li Shen, Keqin Li, and Albert Y Zomaya. Stochastic client selection for federated learning with volatile clients. *IEEE Internet of Things Journal*, 9(20):20055–20070, 2022.

[123] Zhendong Song, Hongguang Sun, Howard H Yang, Xijun Wang, Yan Zhang, and Tony QS Quek. Reputation-based federated learning for secure wireless networks. *IEEE Internet of Things Journal*, 9(2):1212–1226, 2021.

[124] Sebastian Caldas, Jakub Konečny, H Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *arXiv preprint arXiv:1812.07210*, 2018.

[125] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[126] Xubo Yue, Maher Nouiehed, and Raed Al Kontar. Gifair-fl: A framework for group and individual fairness in federated learning. *INFORMS Journal on Data Science*, 2(1):10–23, 2023.

[127] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *International conference on machine learning*, pages 4615–4625. PMLR, 2019.

[128] Wei Du, Depeng Xu, Xintao Wu, and Hanghang Tong. Fairness-aware agnostic federated learning. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 181–189. SIAM, 2021.

[129] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. In *International Conference on Learning Representations*, 2020.

[130] Lingjuan Lyu, Jiangshan Yu, Karthik Nandakumar, Yitong Li, Xingjun Ma, Jiong Jin, Han Yu, and Kee Siong Ng. Towards fair and privacy-preserving federated deep models. *IEEE Transactions on Parallel and Distributed Systems*, 31(11):2524–2541, 2020.

[131] Liang Gao, Li Li, Yingwen Chen, Wenli Zheng, ChengZhong Xu, and Ming Xu. Fifl: A fair incentive mechanism for federated learning. In *Proceedings of the 50th International Conference on Parallel Processing*, pages 1–10, 2021.

[132] Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. A fairness-aware incentive scheme for federated learning. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 393–399, 2020.

[133] Wissam Qassim Al-Salih and Domokos Esztergár-Kiss. Linking mode choice with travel behavior by using logit model based on utility function. *Sustainability*, 13(8):4332, 2021.

[134] Ruggiero Lovreglio, Dino Borri, Luigi dell'Olio, and Angel Ibeas. A discrete choice model based on random utilities for exit choice in emergency evacuations. *Safety Science*, 62:418–426, 2014.

[135] Euphemia L Sibanda, Marc d'Elbée, Galven Maringwa, Nancy Ruhode, Mary Tumushime, Claudius Madanhire, Jason J Ong, Pitchaya Indravudh, Constancia Watadzaushe, Cheryl C Johnson, et al. Applying user preferences to optimize the contribution of hiv self-testing to reaching the "first 90" target of unaids fast-track strategy: results from discrete choice experiments in zimbabwe. *Journal of the International AIDS Society*, 22:e25245, 2019.

[136] Seyyed Hadi Hashemi and Jaap Kamps. Skip or stay: Users' behavior in dealing with onsite information interaction crowd-bias. In *Proceedings of the 2017 Conference on Human Information Interaction and Retrieval*, pages 389–392, 2017.

[137] Miguel Paredes, Erik Hemberg, Una-May O'Reilly, and Chris Zegras. Machine learning or discrete choice models for car ownership demand estimation and prediction? In *IEEE International Conference on Models and Technologies for Intelligent Transportation Systems*, pages 780–785, 2017.

[138] Shahar Ayal and GUY Hochman. Ignorance or integration: The cognitive processes underlying choice behavior. *Journal of Behavioral Decision Making*, 22(4):455–474, 2009.

[139] Saurabh Arora and Prashant Doshi. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297:103500, 2021.

[140] Nathan Sandholtz, Yohsuke Miyamoto, Luke Bornn, and Maurice A Smith. Inverse bayesian optimization: Learning human acquisition functions in an exploration vs exploitation search task. *Bayesian Analysis*, 18(1):1–24, 2023.

[141] Antti Kangasrääsiö, Kumaripaba Athukorala, Andrew Howes, Jukka Corander, Samuel Kaski, and Antti Oulasvirta. Inverse modeling of complex interactive behavior with abc. In *Proceedings of the SIGCHI conference on Human factors in computing systems. ACM*, 2017.

[142] Yuanying Zhao, Jacek Pawlak, and John W Polak. Inverse discrete choice modelling: theoretical and practical considerations for imputing respondent attributes from the patterns of observed choices. *Transportation Planning and Technology*, 41(1):58–79, 2018.

[143] Zhan Zhao and Yuebing Liang. A deep inverse reinforcement learning approach to route choice modeling with context-dependent rewards. *Transportation Research Part C: Emerging Technologies*, 149:104079, 2023.

[144] Zhihua Cui, Xianghua Xu, XUE Fei, Xingjuan Cai, Yang Cao, Wensheng Zhang, and Jinjun Chen. Personalized recommendation system based on collaborative filtering for iot scenarios. *IEEE Transactions on Services Computing*, 13(4):685–695, 2020.

[145] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. Atrank: An attention-based user behavior modeling framework for recommendation. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

[146] Peter J Huber. *Robust statistics*. John Wiley & Sons, 2004.

[147] Anqi Liu, Guanya Shi, Soon-Jo Chung, Anima Anandkumar, and Yisong Yue. Robust regression for safe exploration in control. In *Learning for Dynamics and Control*, pages 608–619. PMLR, 2020.

[148] Peter D Grünwald and A Philip Dawid. Game theory, maximum entropy, minimum discrepancy and robust bayesian decision theory. *The Annals of Statistics*, 32(4):1367–1433, 2004.

[149] Min Yang, Linli Xu, Martha White, Dale Schuurmans, and Yao-liang Yu. Relaxed clipping: A global training method for robust regression and classification. *Advances in Neural Information Processing Systems*, 23, 2010.

[150] Trac D Nguyen, Nam H ansd Tran. Exact recoverability from dense corrupted observations via ℓ_1-minimization. *IEEE Transactions on Information Theory*, 59(4):2017–2035, 2013.

[151] Dong Huang, Ricardo Cabral, and Fernando De la Torre. Robust regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):363–375, 2015.

[152] Yudong Chen, Constantine Caramanis, and Shie Mannor. Robust sparse regression under adversarial corruption. pages 774–782. PMLR, 2013.

[153] Kush Bhatia, Prateek Jain, and Purushottam Kar. Robust regression via hard thresholding. *Advances in Neural Information Processing Systems*, 28, 2015.

[154] Kush Bhatia, Prateek Jain, Parameswaran Kamalaruban, and Purushottam Kar. Consistent robust regression. *Advances in Neural Information Processing Systems*, 30, 2017.

[155] Xiaotong Yuan and Ping Li. Stability and risk bounds of iterative hard thresholding. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 1702–1710. PMLR, 2021.

[156] Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Appl. Comput. Harmon. Anal.*, 27(3):265–274, 2009.

[157] Dimitris Bertsimas, Jack Dunn, Colin Pawlowski, and Ying Daisy Zhuo. Robust classification. *INFORMS Journal on Optimization*, 1(1):2–34, 2019.

[158] Huan Xu, Constantine Caramanis, and Shie Mannor. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10(7), 2009.

[159] Yunlong Feng, Yuning Yang, and et al. Robust support vector machines for classification with nonconvex and smooth losses. *Neural computation*, 28(6):1217–1247, 2016.

[160] Kaan Gokcesu and Hakan Gokcesu. Generalized huber loss for robust learning and its efficient minimization for a robust statistics. *arXiv preprint arXiv:2108.12627*, 2021.

[161] Laurens Maaten, Minmin Chen, Stephen Tyree, and Kilian Weinberger. Learning with marginalized corrupted features. In *Proceedings of the Internal Conference on Machine Learning*, pages 410–418. PMLR, 2013.

[162] Junyuan Hong, Huanhuan Chen, and Feng Lin. Disturbance grassmann kernels for subspace-based learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1521–1530, 2018.

[163] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. *Advances in Neural Information Processing Systems*, 26, 2013.

[164] Jiashi Feng, Huan Xu, Shie Mannor, and Shuicheng Yan. Robust logistic regression and classification. *Advances in Neural Information Processing Systems*, 27, 2014.

[165] Nikola Konstantinov and Christoph Lampert. Robust learning from untrusted sources. In *Proceedings of the Internal Conference on Machine Learning*, pages 3488–3498. PMLR, 2019.

[166] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. LLM-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009, 2023.

[167] Zhe Liu, Chunyang Chen, Junjie Wang, Mengzhuo Chen, Boyu Wu, Xing Che, Dandan Wang, and Qing Wang. Make LLM a testing expert: Bringing human-like interaction to mobile gui testing via functionality-aware decisions. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13, 2024.

[168] Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, et al. Sotopia: Interactive evaluation for social intelligence in language agents. In *International Conference on Learning Representations*, 2023.

[169] Guangzhi Sun, Xiao Zhan, and Jose Such. Building better ai agents: A provocation on the utilisation of persona in LLM-based conversational agents. In *Proceedings of the 6th ACM Conference on Conversational User Interfaces*, pages 1–6, 2024.

[170] Lei Wang, Heyang Gao, Xiaohe Bo, Xu Chen, and Ji-Rong Wen. Yulan-onesim: Towards the next generation of social simulator with large language models. *arXiv preprint arXiv:2505.07581*, 2025.

[171] Jacy Reese Anthis, Ryan Liu, Sean M Richardson, Austin C Kozlowski, Bernard Koch, Erik Brynjolfsson, James Evans, and Michael S Bernstein. Position: LLM social simulations are a promising research method. In *International Conference on Machine Learning (Position Paper Track)*, 2025.

[172] Chaoran Chen, Bingsheng Yao, Yanfang Ye, Dakuo Wang, and Toby Jia-Jun Li. Evaluating the LLM agents for simulating humanoid behavior. In *Proceedings of the 20234 CHI Conference on Human Factors in Computing Systems*, 2024.

[173] Juanjuan Meng. Ai emerges as the frontier in behavioral science. *Proceedings of the National Academy of Sciences*, 121(10):e2401336121, 2024.

[174] Madeleine Grunde-McLaughlin, Michelle S Lam, Ranjay Krishna, Daniel S Weld, and

Jeffrey Heer. Designing LLM chains by adapting techniques from crowdsourcing workflows. *ACM Transactions on Computer-Human Interaction*, 32(3):1–57, 2025.

[175] Jiechen Xu, Lei Han, Shazia Sadiq, and Gianluca Demartini. On the role of large language models in crowdsourcing misinformation assessment. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 18, pages 1674–1686, 2024.

[176] Nagraj Deshmukh, Abithaa Shree Venkatesh, Ann Mathew, Monika Madugula, Pratik Mahesh Merchant, Matthew A Lanham, and Sharan Shirodkar. Harnessing LLMs to build an autonomous marketing agent. In *World Congress in Computer Science, Computer Engineering & Applied Computing*, pages 271–282. Springer, 2024.

[177] Leng Cai, Junxuan He, Yikai Li, Junjie Liang, Yuanping Lin, Ziming Quan, Yawen Zeng, and Jin Xu. RTBAgent: A LLM-based agent system for real-time bidding. In *The ACM Web Conference*, pages 104–113, 2025.

[178] Yubo Shu, Haonan Zhang, Hansu Gu, Peng Zhang, Tun Lu, Dongsheng Li, and Ning Gu. Rah! recsys–assistant–human: A human-centered recommendation framework with LLM agents. *IEEE Transactions on Computational Social Systems*, 11(5):6759–6770, 2024.

[179] Ivens Da Silva Portugal, Paulo Alencar, and Donald Cowan. An agentic ai-based multi-agent framework for recommender systems. In *2024 IEEE International Conference on Big Data*, pages 5375–5382, 2024.

[180] Yuchen Zhang, Xi Chen, Dengyong Zhou, and Michael I Jordan. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. *Advances in Neural Information Processing Systems*, 27, 2014.

[181] Huichuan Xia and Brian McKernan. Privacy in crowdsourcing: a review of the threats and challenges. *Computer Supported Cooperative Work (CSCW)*, 29:263–301, 2020.

[182] Max H Sims, Margie Hodges Shaw, Seth Gilbertson, Joseph Storch, and Marc W Halterman. Legal and ethical issues surrounding the use of crowdsourcing among healthcare providers. *Health informatics journal*, 25(4):1618–1630, 2019.

[183] Deniz Iren and Semih Bilgen. Cost of quality in crowdsourcing. *Human Computation*, 1(2), 2014.

[184] Haoyu Xie, Eddy Maddalena, Rehab Qarout, and Alessandro Checco. The dark side of recruitment in crowdsourcing: Ethics and transparency in micro-task marketplaces. *Computer Supported Cooperative Work*, 32(3):439–474, 2023.

[185] Meysam Alizadeh, Fabrizio Gilardi, Zeynab Samei, and Mohsen Mosleh. Web-browsing LLMs can access social media profiles and infer user demographics. *arXiv preprint arXiv:2507.12372*, 2025.

[186] Yubo Wang, Min Tang, Nuo Shen, Shujie Cui, and Weiqing Wang. Privacy risks of LLM-empowered recommender systems: An inversion attack perspective. In *Proceedings of the 19th ACM Conference on Recommender Systems*, pages 812–821, 2025.

[187] Jacy Anthis, Sean Richardson, Austin Kozlowski, Bernard Koch, Erik Brynjolfsson, James Evans, and Michael Bernstein. Position: LLM social simulations are a promising research method. In *International Conference on Machine Learning*, 2025.

[188] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[189] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[190] Team Gemini, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[191] Team Gemma, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report, 2025.

[192] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

[193] Jiaxiang Li, Siliang Zeng, Hoi-To Wai, Chenliang Li, Alfredo Garcia, and Mingyi Hong. Getting more juice out of the sft data: Reward learning from human demonstration improves sft for LLM alignment. *Advances in Neural Information Processing Systems*, 2014.

[194] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023.

[195] Ermo Hua, Biqing Qi, Kaiyan Zhang, Yue Yu, Ning Ding, Xingtai Lv, Kai Tian, and Bowen Zhou. Intuitive fine-tuning: Towards unifying sft and rlhf into a single process. *arXiv preprint arXiv:2405.11870*, 2024.

[196] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[197] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[198] Yuchen Xia, Jiho Kim, Yuhan Chen, Haojie Ye, Souvik Kundu, Cong Hao, and Nishil Talati. Understanding the performance and estimating the cost of LLM fine-tuning. *arXiv preprint arXiv:2408.04693*, 2024.

[199] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[200] Yilun Liu, Shimin Tao, Xiaofeng Zhao, Ming Zhu, Wenbing Ma, Junhao Zhu, Chang Su, Yutai Hou, Miao Zhang, Min Zhang, et al. Coachlm: Automatic instruction revisions improve the data quality in LLM instruction tuning. In *2024 IEEE 40th International Conference on Data Engineering*, pages 5184–5197, 2024.

[201] Min-Hsuan Yeh, Leitian Tao, Jeffrey Wang, Xuefeng Du, and Yixuan Li. How reliable is human feedback for aligning large language models? *arXiv preprint arXiv:2410.01957*, 2024.

[202] Ting-Yun Chang and Robin Jia. Data curation alone can stabilize in-context learning. *arXiv preprint arXiv:2212.10378*, 2022.

[203] Xiao-Kun Wu, Min Chen, Wanyi Li, Rui Wang, Limeng Lu, Jia Liu, Kai Hwang, Yixue Hao, Yanru Pan, Qingguo Meng, et al. LLM fine-tuning: Concepts, opportunities, and challenges. *Big Data and Cognitive Computing*, 9(4):87, 2025.

[204] Fangcong Yin, Xi Ye, and Greg Durrett. LoFIT: Localized fine-tuning on LLM representations. *Advances in Neural Information Processing Systems*, 37:9474–9506, 2024.

[205] Guang Zhao, Byung-Jun Yoon, Gilchan Park, Shantenu Jha, Shinjae Yoo, and Xiaoning Qian. Pareto prompt optimization. In *International Conference on Learning Representations*, 2025.

[206] J Diego Zamfirescu-Pereira, Richmond Y Wong, Bjoern Hartmann, and Qian Yang. Why Johnny can't prompt: how non-AI experts try (and fail) to design LLM prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–21, 2023.

[207] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. Neural prompt search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(7):5268–5280, 2024.

[208] Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. Understanding the effects of rlhf on LLM generalisation and diversity. *arXiv preprint arXiv:2310.06452*, 2023.

[209] Andrew J Peterson. Ai and the problem of knowledge collapse. *arXiv preprint arXiv:2404.03502*, 2024.

[210] Nabeel Seedat, Nicolas Huynh, Boris Van Breugel, and Mihaela Van Der Schaar. Curated LLM: Synergy of LLMs and data curation for tabular augmentation in low-data regimes. In *International Conference on Machine Learning*, pages 44060–44092, 2024.

[211] Weijia Xu, Nebojsa Jojic, Sudha Rao, Chris Brockett, and Bill Dolan. Echoes in ai: Quantifying lack of plot diversity in LLM outputs. *arXiv preprint arXiv:2501.00273*, 2024.

[212] Mingzhang Yin and Mingyuan Zhou. Semi-implicit variational inference. In *International Conference on Machine Learning*, pages 5660–5669, 2018.

[213] Biraj Dahal, Alexander Havrilla, Minshuo Chen, Tuo Zhao, and Wenjing Liao. On deep generative models for approximation and estimation of distributions on manifolds. *Advances in Neural Information Processing Systems*, 35:10615–10628, 2022.

[214] Eddie Aamari, Jisu Kim, Frédéric Chazal, Bertrand Michel, Alessandro Rinaldo, and Larry Wasserman. Estimating the reach of a manifold. *Electronic Journal of Statistics*, 13(1):1359–1399, 2019.

[215] Danny Wood, Tingting Mu, Andrew M Webb, Henry WJ Reeve, Mikel Luján, and Gavin Brown. A unified theory of diversity in ensemble learning. *Journal of machine learning research*, 24(359):1–49, 2023.

[216] Anastasios N Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I Jordan, and Tijana Zrnic. Prediction-powered inference. *Science*, 382(6671):669–674, 2023.

[217] Lionel S Penrose. The elementary statistics of majority voting. *Journal of the Royal Statistical Society*, 109(1):53–57, 1946.

[218] Jean-Marie Blin and Andrew B Whinston. Discriminant functions and majority voting. *Management Science*, 21(5):557–566, 1975.

[219] Jennifer Wortman Vaughan. Making better use of the crowd: How crowdsourcing can advance machine learning research. *Journal of Machine Learning Research*, 18(193):1–46, 2018.

[220] Folasade Olubusola Isinkaye, Yetunde O Folajimi, and Bolande Adefowoke Ojokoh. Recommendation systems: Principles, methods and evaluation. *Egyptian informatics journal*, 16(3):261–273, 2015.

[221] Joshua C Yang, Carina I Hausladen, Dominik Peters, Evangelos Pournaras, Regula Hnggli Fricker, and Dirk Helbing. Designing digital voting systems for citizens: Achieving fairness and legitimacy in participatory budgeting. *Digital Government: Research and Practice*, 5(3):1–30, 2024.

[222] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier Movellan, and Paul Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in Neural Information Processing Systems*, 22, 2009.

[223] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The Llama 3 herd of models. *arXiv preprint arxiv:2407.21783*, 2024.

[224] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report, 2025.

[225] Adji B Dieng, Yoon Kim, Alexander M Rush, and David M Blei. Avoiding latent variable collapse with generative skip models. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2397–2405, 2019.

[226] Zach Nussbaum, John X Morris, Brandon Duderstadt, and Andriy Mulyar. Nomic embed: Training a reproducible long context text embedder. *arXiv preprint arXiv:2402.01613*, 2024.

[227] Diederik P Kingma. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2014.

[228] Ang Li, Haozhe Chen, Hongseok Namkoong, and Tianyi Peng. LLM generated persona is a promise with a catch. *arXiv preprint arXiv:2503.16527*, 2025.

[229] Tiancheng Hu and Nigel Collier. Quantifying the persona effect in LLM simulations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 10289–10307, 2024.

[230] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

[231] Ollie Liu, Deqing Fu, Dani Yogatama, and Willie Neiswanger. Dellma: Decision making under uncertainty with large language models. In *International Conference on Learning Representations*, 2024.

[232] Carlos Olea, Holly Tucker, Jessica Phelan, Cameron Pattison, Shen Zhang, Maxwell Lieb, Doug Schmidt, and Jules White. Evaluating persona prompting for question

answering tasks. In *International Conference on Artificial Intelligence and Soft Computing*, 2024.

[233] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

[234] Jiaxin Pei and David Jurgens. When do annotator demographics matter? measuring the influence of annotator demographics with the POPQUORN dataset. In *Proceedings of the 17th Linguistic Annotation Workshop*, pages 252–265, 2023.

[235] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation, 2024.

[236] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52, 2015.

[237] Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pages 21246–21263, 2024.

[238] Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. In *International Conference on Machine Learning*, 2026.

[239] Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. *Advances in Neural Information Processing Systems*, 7, 1994.

[240] Ying Lin, Shan Liu, and Shuai Huang. Selective sensing of a heterogeneous population of units with dynamic health conditions. *IISE Transactions*, 50(12):1076–1088, 2018.

[241] Xi Zhu, Jingshuo Feng, Shuai Huang, and Cynthia Chen. An online updating method for time-varying preference learning. *Transportation Research Part C: Emerging Technologies*, 121:102849, 2020.

[242] Jiahao Chen, Nathan Kallus, Xiaojie Mao, Geoffry Svacha, and Madeleine Udell. Fairness under unawareness: Assessing disparity when protected class is unobserved. In *FAccT*, 2019.

[243] Alekh Agarwal, Miroslav Dudík, and Zhiwei Steven Wu. Fair regression: Quantitative definitions and reduction-based algorithms. In *International Conference on Machine Learning*, 2019.

[244] Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. Counterfactual fairness. *Advances in Neural Information Processing Systems*, 2017.

[245] Dana Pessach and Erez Shmueli. Algorithmic fairness. *arXiv preprint arXiv:2001.09784*, 2020.

[246] Misha Belkin, Partha Niyogi, and Vikas Sindhwani. On manifold regularization. In *IWSMAI*, 2005.

[247] Jianfeng Chi, Yuan Tian, Geoffrey J Gordon, and Han Zhao. Understanding and mitigating accuracy disparity in regression. In *International conference on machine learning*, pages 1866–1876. PMLR, 2021.

[248] Shengyuan Hu, Zhiwei Steven Wu, and Virginia Smith. Provably fair federated learning via bounded group loss. *arXiv preprint arXiv:2203.10190*, 2022.

[249] Tianlin Li, Zhiming Li, Anran Li, Mengnan Du, Aishan Liu, Qing Guo, Guozhu Meng, and Yang Liu. Fairness via group contribution matching. In *IJCAI*, pages 436–445, 2023.

[250] Ruicheng Xian, Lang Yin, and Han Zhao. Fair and optimal classification via post-processing. In *International Conference on Machine Learning*, pages 37977–38012. PMLR, 2023.

[251] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[252] Madeline Navarro, Camille Little, Genevera I Allen, and Santiago Segarra. Data augmentation via subgroup mixup for improving fairness. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7350–7354. IEEE, 2024.

[253] Xi Zhu, Feilong Wang, Cynthia Chen, and Derek D Reed. Personalized control for promoting sustainable travel behaviors. *Transportation Research Procedia*, 38:730–750, 2019.

[254] Xi Zhu, Feilong Wang, Cynthia Chen, and Derek D Reed. Personalized incentives for promoting sustainable travel behaviors. *Transportation Research Part C: Emerging Technologies*, 113:314–331, 2020.

[255] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management*, pages 1–9. IEEE, 2008.

[256] Jean-Loup Loyer, Elsa Henriques, and Steve Wiseall. Comparison of binary classifiers for data-driven prognosis of jet engines health. In *PHM Society European Conference*, volume 2, 2014.

[257] Yacine Aribi, Saliha Boutora, Pr Razika Boushaki Zamoum, Hafid Menasria, Abdelkader Abdellaoui, and Pr Abdellah Kouzou. Turbofan engine rul prediction using ica and machine learning algorithms. In *2023 20th International Multi-Conference on Systems, Signals & Devices (SSD)*, pages 477–484. IEEE, 2023.

[258] Fernando Costabella, Keval B Patel, Adedimeji V Adepoju, Purnima Singh, Hussein Attia Hussein Mahmoud, Awais Zafar, Tirath Patel, Ninad A Watekar, Navya Mallesh, Moiz Fawad, et al. Healthcare cost and outcomes associated with surgical site infection and patient outcomes in low-and middle-income countries. *Cureus*, 15(7), 2023.

[259] Yann LeCun, Lawrence D Jackel, Léon Bottou, Corinna Cortes, John S Denker, Harris Drucker, Isabelle Guyon, Urs A Muller, Eduard Sackinger, Patrice Simard, et al. Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural networks: the statistical mechanics perspective*, 261(276):2, 1995.

[260] Matthias De Lange, Xu Jia, Sarah Parisot, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Unsupervised model personalization while preserving privacy and scalability: An open problem. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14463–14472, 2020.

[261] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016.

[262] Alejandro G Martín, Alberto Fernández-Isabel, Isaac Martín de Diego, and Marta Beltrán. A survey for user behavior analysis based on machine learning techniques: current models and applications. *Applied Intelligence*, 51(8):6029–6055, 2021.

[263] Hong-Bin Yan and Ming Li. An uncertain kansei engineering methodology for behavioral service design. *IISE Transactions*, 53(5):497–522, 2021.

[264] Cao Xiao, Shupeng Gui, Ji Liu, Yu Cheng, Xiaoning Qian, and Shuai Huang. Longitudinal planning for personalized health management using daily behavioral data. *IISE Transactions Healthc. Syst. Eng.*, 9(3):243–249, 2019.

[265] Md Rafiqul Islam, Imran Razzak, Xianzhi Wang, Peter Tilocca, and Guandong Xu. Ucbvis: understanding customer behavior sequences with visual interactive system.

In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2021.

[266] Xiaohui Xie, Jiaxin Mao, Maarten de Rijke, Ruizhe Zhang, Min Zhang, and Shaoping Ma. Constructing an interaction behavior model for web image search. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 425–434, 2018.

[267] Pengfei Wang, Chi Lin, Mohammad S Obaidat, Zhen Yu, Ziqi Wei, and Qiang Zhang. Contact tracing incentive for covid-19 and other pandemic diseases from a crowdsourcing perspective. *IEEE Internet of Things Journal*, 8(21):15863–15874, 2021.

[268] Jian Sun, Jiyan Wu, Feng Xiao, Ye Tian, and Xiangdong Xu. Managing bottleneck congestion with incentives. *Transportation Research Part B: Methodological*, 134:143–166, 2020.

[269] Daniel McFadden et al. Conditional logit analysis of qualitative choice behavior. 1973.

[270] Rui Zhong and Sandra L Robinson. What happens to bad actors in organizations? a review of actor-centric outcomes of negative behavior. *Journal of Management*, 47(6):1430–1467, 2021.

[271] Merrill Warkentin and Robert Willison. Behavioral and policy issues in information systems security: the insider threat. *European Journal of Information Systems*, 18(2):101–105, 2009.

[272] Ujwal Gadiraju, Ricardo Kawase, Stefan Dietze, and Gianluca Demartini. Understanding malicious behavior in crowdsourcing platforms: The case of online surveys. In *Proceedings of the Annual ACM Conference on Human Factors in Computing Systems*, pages 1631–1640, 2015.

[273] Carsten Eickhoff and Arjen P de Vries. Increasing cheat robustness of crowdsourcing tasks. *Information Retrieval*, 16(2):121–137, 2013.

[274] Sebastian Böttcher, Solveig Vieluf, Elisa Bruno, Boney Joseph, Nino Epitashvili, Andrea Biondi, Nicolas Zabler, Martin Glasstetter, Matthias Dümpelmann, Kristof Van Laerhoven, et al. Data quality evaluation in wearable monitoring. *Scientific reports*, 12(1):21412, 2022.

[275] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. *Advances in neural information processing systems*, 30, 2017.

[276] Jiahui Yu, Kun Wang, Peng Li, Rui Xia, Song Guo, and Minyi Guo. Efficient trustworthiness management for malicious user detection in big data collection. *IEEE Transactions on Big Data*, 2017.

[277] Michael Luca and Georgios Zervas. Fake it till you make it: Reputation, competition, and yelp review fraud. *Management Science*, 62(12):3412–3427, 2016.

[278] Lin Yang, Mohammad Hassan Hajiesmaili, Mohammad Sadegh Talebi, John CS Lui, Wing Shing Wong, et al. Adversarial bandits with corruptions: Regret lower bound and no-regret algorithm. In *Advances in Neural Information Processing Systems*, 2020.

[279] Yuansi Chen, Armeen Taeb, and Peter Bühlmann. A look at robustness and stability of ℓ_1-versus ℓ_0-regularization: Discussion of papers by bertsimas et al. and hastie et al. *Statistical Science*, 35(4):614–622, 2020.

[280] Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *International Conference on Machine learning*, page 116, 2004.

[281] Battista Biggio, Blaine Nelson, and Pavel Laskov. Support vector machines under adversarial label noise. In *Asian Conference on Machine Learning*, pages 97–112. PMLR, 2011.

[282] Yibo Wang and Rohana J Karunamuni. High-dimensional robust regression with l_q-loss functions. *Computational Statistics & Data Analysis*, 176:107567, 2022.

[283] Junnan Li, Richard Socher, and Steven C.H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, 2020.

[284] Taehyeon Kim, Jongwoo Ko, JinHwan Choi, Se-Young Yun, et al. Fine samples for learning with noisy labels. *Advances in Neural Information Processing Systems*, 34:24137–24149, 2021.

[285] Shenda Hong, Yanbo Xu, Alind Khare, Satria Priambada, Kevin Maher, Alaa Aljiffry, Jimeng Sun, and Alexey Tumanov. Holmes: health online model ensemble serving for deep learning models in intensive care units. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1614–1624, 2020.

[286] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

[287] Ninareh Mehrabi, Muhammad Naveed, Fred Morstatter, and Aram Galstyan. Exacerbating algorithmic bias through fairness attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8930–8938, 2021.

[288] David Solans, Battista Biggio, and Carlos Castillo. Poisoning attacks on algorithmic fairness. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 162–177. Springer, 2020.