University of Southampton

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Multi-Advisor Sequential Decision-Making Without Ground Truth

by

Zhaori Guo

ORCiD: 0000-0002-1957-7059

A thesis for the degree of
Doctor of Philosophy

November 2023

University of Southampton

Abstract

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Doctor of Philosophy

Multi-Advisor Sequential Decision-Making Without Ground Truth

by Zhaori Guo

Decision-making from potentially unreliable advice is an important problem in many settings, such as lending, investment, ensemble machine learning, and crowd-sourcing. In such settings, advice can often be elicited from multiple advisers and aggregated to make a more reliable decision, especially when the decisions have important consequences. In addition, often, similar decisions are made over time using the same set of advisers. Therefore, the reliability or *trustworthiness* of advisers can be utilized to improve decision accuracy and learned and updated over time. However, this is challenging especially when there is no access to the *ground truth*, i.e., when there is no information about the true or ideal decision, even after the fact, or this information is only available after a considerable delay (e.g., in the case of a loan default). While there is extensive work in decision-making from multiple advisers, existing work focuses on single-shot static decision-making, and does not account for the sequential nature of decisions. To address this gap, this thesis addresses settings where multiple decisions are made sequentially over time, without access to the ground truth, and where we have no prior information about advisors' trustworthiness. We refer to this as the *multi-advisor sequential decision-making problem*.

To address this problem, first, we propose the Multi-Advisor Binary Sequential Decision-Making method (MABSDM). In this setting, a decision-maker needs to make decisions on a sequence of *problems*, which includes the essential factors for making decisions. For each problem, a set of advisors provides advice between binary options and the decision-maker needs to aggregate their advice to make a decision. To be specific, MABSDM (1) models the advisors' trustworthiness sequentially without prior information, (2) makes optimal decisions from the advice and trustworthiness of multiple imperfect advisors without ground truth. In addition, our results show that MABSDM has higher decision accuracy than benchmarks using state-of-the-art models including Bayesian aggregation, weighted voting, and Beta distribution trustworthiness model. Moreover, MABSDM outperforms benchmarks in terms of modeling the trustworthiness of advisors in most results.

Second, we then apply MABSDM to an interactive reinforcement learning setting whereby proposing a method named Multi-Advisor Interactive Reinforcement Learning system (MAIRL). In more detail, interactive reinforcement learning is an effective way to accelerate agent learning by feedback from human advisors to agents. However, if the human advisor is not always reliable, it often hinders the agent's training. To address this problem, we introduce multiple advisors to turn this problem into a multi-advisor binary sequential decision-making problem. Specifically, in MAIRL, we use MABSDM to aggregate the binary feedback from multiple imperfect advisors into a reliable reward for agent training in a reward-sparse environment. In addition, the review model in MAIRL can correct the unreliable reward from advisors. In particular, our experiments for evaluating feedback forms show that the binary feedback outperforms other feedback forms including ranking feedback, scaling feedback, and state value feedback. Finally, we conduct grid-world experiments to show that the policy trained by the MAIRL with the review model is closer to the optimal policy than that without a review model.

Third, we propose a utility maximization method based on MABSDM, namely Multi-Advisor Dynamic Decision-Making (MADDM). In more detail, in practice, making a correct decision often has great rewards while a failed decision has a significant cost, and gathering advice from a set of advisors has a cost. We take into account balancing the value of decisions and the cost associated with querying advisors in the multi-advisor binary sequential decision-making problem. Therefore, the challenge is finding an advisor selection strategy that retrieves reliable advice and maximizes the overall utility, which is the expected return of the decision-making. To address this challenge, MADDM considers selecting advisors by balancing the advisors' costs, advisors' trustworthiness, and the value of the problem and then using MABSDM to make the optimal decision. Moreover, we evaluate our algorithm through several numerical experiments. The results show that our approach outperforms two other methods that combine state-of-the-art models.

Finally, we extend MABSDM to a general method, namely Multi-Advisor Sequential Decision-Making (MASDM), which can make decisions among multiple options, not just binary options. In addition, we evaluate MASDM through extensive experiments in simulated environments. Moreover, we apply our method to ensemble machine learning using the experiments by the MNIST database. The results show that MASDM has better decision accuracy and the ability to trustworthiness assessment than the five benchmarks that use state-of-the-art methods, achieving a maximum improvement of 22% in accuracy compared to Bayesian aggregation methods.

Contents

List of Figures

List of Tables

Declaration of Authorship

I declare that this thesis and the work presented in it is my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Parts of this work have been published as:
   Guo, Z., Norman, T. J., Gerding, E. H. (2022, November). MTIRL: Multi-trainer Interactive Reinforcement Learning System. In International Conference on Principles and Practice of Multi-Agent Systems (pp. 227-242).
   Guo, Z., Norman, T. J., Gerding, E. H. (2023, May). MADDM: Multi-Advisor Dynamic Binary Decision-Making by Maximizing the Utility. In Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems.
   Guo, Z. (2023, May). Multi-Advisor Dynamic Decision Making. In Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems.

Signed:.. Date:..................

Acknowledgements

I would like to express my deepest gratitude to all those who have helped and supported me during my Ph.D. study.

First and foremost, I would like to thank my supervisors, Professor Timothy J. Norman and Professor Enrico H. Gerding, for their invaluable guidance, patience, and encouragement throughout my research. Their immense knowledge and plentiful experience have inspired me in my academic pursuit.

In addition, special thanks go to my parents, for their unconditional love and support both financially and emotionally through this challenging journey. I would not have been able to complete my studies without their unwavering belief in me.

Finally, I'd like to thank all the people who have supported me along the way with their advice, help or love. You have made my experience fulfilling and memorable.

Definitions and Abbreviations

X	advisor set
x	advisor index
T	problem ordered set
t	problem index
pos	positive option
neg	negative option
$A_{Y_t}^{pos}$	set of advisors who choose the option pos
$A_{Y_t}^{neg}$	set of advisors who choose the option neg
Y_t	set of advisors who give advice to the problem t
α_t^x	correct estimated evidence of the advisor x
β_t^x	wrong estimated evidence of the advisor x
θ_t^x	uncertainty of the advisor x
τ_t^x	trustworthiness of the advisor x
f	decision-making function
d_t	decision of the problem t
d_t^*	ground truth of the problem t
i_t	confidence value of the problem t
P_t^{pos}	probability that $pos = d_t^*$
P_t^{neg}	probability that $neg = d_t^*$
r_t'	human reward of the state-action pair at time t
r_t^*	correct reward of the state-action pair at time t
l	learning rate of reinforcement learning
γ	reward discount coefficient γ
τ_x'	trustworthiness of the advisor x from Beta Sampling
f_s	selection function
c_x	price of the advisor x
u_t	utility of problem t
v_t^+	profits that if $d_t = d_t^*$
v_t^-	loss that if $d_t \neq d_t^*$
O	option set
o	option index
$A_{Y_t}^{o}$	the set of advisors who choose the option o

i_t^o	confidence value of the option o of the problem t
P_t^o	probability that the option $o = d_t^*$
MABSDM	Multi-Advisor Binary Sequential Decision-Making
BBWVE	Binary Bayesian and Weighted Voting Ensemble
BCT	Binary Cautious Trustworthiness
WV	Weighted Voting
BYS	Bayesian Aggregation
ERGd	Extended Rectified Gaussian distribution
Beta	Beta Distribution Trustworthiness Model
RL	Reinforcement Learning
IRL	Interactive Reinforcement Learning
SARSA	State-Action-Reward-State-Action
MAIRL	Multi-Advisor Interactive Reinforcement Learning
MADDM	Multi-Advisor Dynamic Decision-Making
FNA	Fixed Number of Advisors
BC	Budget-Constraint
MABDM	Multi-Advisor Sequential Decision-Making
BWVE	Bayesian and Weighted Voting Ensemble
CT	Cautious Trustworthiness
TAMER	Training an Agent Manually via Evaluative Reinforcement
VI-TAMER	Value Iteration TAMER
DQN-TAMER	Deep Q Network TAMER
UCB	Upper Confidence Bound Algorithm

Chapter 1

Introduction

In this chapter, we provide a comprehensive overview of our research. First, we explain the motivation for our research on sequential decision-making in Section 1.1. Second, we introduce the current challenges associated with sequential decision-making in Section 1.2. Third, we establish our research objectives in Section 1.3, highlighting the goals we aim to achieve. Fourth, Section 1.4 introduces our contributions, outlining the methods we propose for addressing the problems in sequential decision-making. Finally, we offer a structure of the thesis in Section 1.5.

1.1 Motivation

The wisdom of crowds possesses extraordinary potential when diverse individuals come together to make decisions. An intriguing instance occurred in 1906 at the Plymouth Country Fair, where 800 participants engaged in a competition to estimate the weight of a cow. The statistician Francis Galton observed that the guessed median of 1207 lbs was within 1% of the true weight of 1198 lbs (Galton, 1907). This example highlights the power of the wisdom of crowds, where aggregating opinions, perspectives, and knowledge from different advisors can lead to surprisingly accurate results, even if individuals' knowledge is limited. An explanation for this phenomenon is that the collective decision of multiple individuals reduces individual biases (Yi et al., 2012). Throughout the thesis, we refer to individuals providing their advice for decision-making problems as *advisors*.

Nowadays, aggregating advice from multiple advisors still is an important approach for decision-making, especially in settings where there is no access to the ground truth. For example, in loan approval, we have no way to know for sure whether the applicant will repay the loan on time, so loan reviewers need to aggregate the advice of multiple advisors to decide whether to agree with the applicant's loan application. Similarly, in America, approximately $150,000$ cases are tried each year by juries, where members vote to determine whether a defendant is guilty (Barkan and Bryjak, 2011). In addition, multi-advisor decision-making is not only applicable to human decision-making but is also significant for ensemble machine learning (Zhou, 2012; Kuncheva, 2014), where multiple machine learning methods are combined to improve predictive performance. For example, *Random Forests*, an ensemble machine learning algorithm, enhances classification accuracy by aggregating the advice of multiple decision trees (Breiman, 2001). Additionally, in interactive reinforcement learning, a method for accelerating agents' training by providing feedback from human advisors, human feedback is not always reliable. If the feedback accuracy of a single human advisor cannot meet the requirements of the agent's learning, introducing multiple advisors and aggregating their feedback to improve the accuracy is effective.

However, it is unrealistic to assume that all advisors have the same level of knowledge, so taking advisors' reliability or *trustworthiness* into decision-making considerations is an effective way to improve the accuracy of decision-making (Zheng et al., 2017; Chen et al., 2022b). One example is crowdsourcing, which infers the true label of tasks by aggregating the advice of multiple advisors. They model and utilize the advisors' trustworthiness to increase the accuracy of decisions (Zheng et al., 2017). Another example is the boosting method in ensemble machine learning. This method allocates the weights for the weak classifiers to aggregate the advice by weighted voting for improving the performance of decision-making (Freund et al., 1996). Nevertheless, such works often assume that the prior information on advisors' trustworthiness is accessible, but we often cannot obtain any prior information in practice. For example, in crowdsourcing, it is difficult for a decision-maker to know the expertise of new advisors. Similarly, internet companies often improve their products by aggregating the feedback from customers, but they cannot determine whether a new user will provide malicious comments. Therefore, it is crucial to develop models for assessing advisors' trustworthiness even in the absence of prior information.

In addition, decision-making is often sequential in practice, i.e., the decision-maker is required to make decisions one after another on a sequence of *problems*, which include the essential factors for making decisions. For instance, bank employees sequentially evaluate multiple loan applications, while doctors also diagnose multiple patients during their consultations one by one. Therefore, these scenarios consider sequentially making optimal decisions without *ground truth*, i.e., when there is no information about the true or ideal decision, even after the fact, or this information is only available after a considerable delay (e.g. in the case of a loan default). Additionally, in practice, problems often hold value, and querying advisors is associated with a cost. Therefore, finding an advisor selection strategy that retrieves reliable advice and maximizes the overall profits or *utility* is a challenging problem. However, previous studies do not give targeted methods to address the above problems (see more details in Chapter 2). These challenges, revolving around sequential decision-making are the main object of study of this thesis and will be discussed in detail next.

1.2 Research Challenges

As highlighted in the preceding section, sequential decision-making holds a pivotal role across numerous domains. Although considerable literature about decision-making has been published over the past decades, many gaps persist in our understanding of sequential decision-making (more details see Chapter 2). This thesis addresses a subset of challenges in multi-advisor sequential decision-making, with a specific emphasis on scenarios without ground truth. Specifically, the research challenges addressed in this thesis are as follows:

1.2.1 Making Optimal Decisions Sequentially without Ground Truth

Optimal decision-making is an extensively discussed topic in various literature. However, existing methods frequently fall short of addressing the challenges in sequential decision-making (see more details in Section 2.1). One key challenge involves making optimal decisions in the absence of ground truth. For instance, whether a patient will recover or not can be only based on statistical evidence of previous similar scenarios. Hence, it becomes crucial to develop methods that can infer the ground truth for decision-making.

In addition, another challenge is to develop online methods for decision-making. In more detail, existing approaches often focus on making decisions by offline methods, whereby the methods process the entire dataset (including all advice of all problems) at once (Kittler et al., 1998; Zhou, 2012; Kuncheva, 2014; Kim and Ghahramani, 2012; Venanzi et al., 2014; Li et al., 2019). In such works, the decision-making is often based on relatively reliable advisors' trustworthiness because the model can get the advice of hundreds of problems for estimating the advisors' trustworthiness (see Section 2.1.3). In contrast, in sequential decision-making problems, the challenge is making reliable decisions according to the changing trustworthiness, especially when the trustworthiness is unreliable (e.g., at the beginning of the sequential decision-making or the initialization stage of modeling new advisors' trustworthiness).

Moreover, the offline method often cannot adapt to new data as it arrives and requires significant computing power and time for training, which causes additional costs and destroys the timeliness of new data. In contrast, online methods are better suited for sequential decision-making as they process data incrementally, enabling adaptive learning from new information in real time.

1.2.2 Modeling Advisors' Trustworthiness without Prior Information

Taking advisors' trustworthiness into consideration during the decision-making process has been shown to enhance decision accuracy (Zheng et al., 2017; Chen et al., 2022b) (see more details in Section 2.2). For example, in weighted voting, advisors' trustworthiness is taken into account, assigning higher weights to more reliable advisors and lower weights to less reliable ones. This approach ensures that decisions are influenced more by trustworthy advisors, leading to more accurate and informed outcomes compared to the majority voting method. However, obtaining prior information about advisors' trustworthiness is often infeasible, so we need to build the trustworthiness without prior information.

In addition, without ground truth, we have no accurate way to evaluate the performance of advisors. If the decision is wrong, updating advisors' trustworthiness based on that wrong decision can lead to increasingly misleading future decisions. Therefore, a carefully designed trustworthiness modeling strategy is necessary because it can have a profound impact on subsequent decisions.

1.2.3 Imperfect Advisors in Interactive Reinforcement Learning

Reinforcement Learning (RL) is a machine learning method to train an agent to make sequential decisions in an environment to maximize cumulative rewards. However, one common challenge in RL is the sparsity of rewards. In many RL scenarios, the agent only receives a reward at the end of an episode or upon reaching specific states within the environment. This limited availability of rewards slows down the agent's exploration process, as it takes a substantial amount of time for the agent to receive feedback on its actions in all critical states of the environment (Arakawa et al., 2018). Therefore, reward sparsity limits the application scenarios of RL (Knox and Stone, 2008b; MacGlashan et al., 2017).

One approach to solve this problem is through *interactive reinforcement learning* (IRL), where human advisors participate in the training process by providing rewards to the agent (see more details in Section 2.3). Research has shown that human rewards can accelerate the learning process of the agent (Knox and Stone, 2008b; MacGlashan et al., 2017). Previous studies have primarily focused on the interaction between an agent and a single human advisor (Knox and Stone, 2008b; MacGlashan et al., 2017; Knox and Stone, 2010). However, a key limitation of relying on a single advisor is the requirement for a perfect advisor, whose reward is always correct. In situations where the quality of rewards from a single advisor is not ideal, it can actually hinder the agent's learning instead of aiding it (Kurenkov et al., 2020).

Therefore, to mitigate the risk of unreliable feedback, employing multiple advisors becomes advantageous. By aggregating the advice of multiple advisors, the challenges associated with unreliable feedback can be addressed. Specifically, in multiple advisors IRL, at each time step, each advisor provides a reward and then a decision-maker aggregates the advice of advisors to make a decision of what reward should be assigned to the agent. This scenario includes decision-making in a time sequence, aggregating advice of multiple advisors, and modeling the trustworthiness of advisors over time. Consequently, multi-advisor IRL is a sequential decision-making problem.

1.2.4 Selecting Advisors by Maximizing the Utility

In practical decision-making scenarios, problems may vary in importance or value. For example, deciding on a $10 million investment carries greater importance compared to choosing what to have for dinner. In addition, querying different advisors may incur distinct costs. For instance, querying an expert often involves a higher cost than querying a layman. Therefore, a crucial problem is the selection of advisors. Specifically, if an insufficient number of advisors is selected, the aggregated advice may lack the desired accuracy. In contrast, selecting all available advisors may lead to excessive costs that outweigh the benefits gained from the problem.

In addition, advisors often possess different levels of expertise, which directly impact the quality of their responses. Given the value of problems, advisors' trustworthiness, and the cost associated with advisors, it becomes essential to consider how many advisors to consult and whom to ask in order to make informed decisions. This can be considered as a multi-armed bandit problem. In this problem, a decision-maker is faced with a set of options (referred to as "arms" or "bandits"), each associated with an unknown probability distribution of rewards, which can only be gradually learned by taking actions over time. At each time step, the decision-maker needs to take an action that selects a unique arm. The objective of the decision-maker is to maximize the cumulative rewards over a series of sequential actions. Similarly, the "action" in our problem is to select the advisors (arms) based on the price and present trustworthiness of advisors and the decision's value. In addition, we need to model advisors' trustworthiness without prior information. Therefore, the challenge is to balance exploiting the current reliable advisors and exploring potential advisors with high trustworthiness. Finally, our aim is to maximize the cumulative utility (maximize the cumulative rewards). However, previous works often consider advisor selection limited by a fixed budget constraint (Tran-Thanh et al., 2012, 2014; Xia et al., 2015; Zhou and Tomlin, 2018;

Cayci et al., 2020), but such works do not consider that decisions might have different values and costs associated with getting them wrong. Accordingly, in sequential decision-making, the challenge is finding an optimal advisor selection strategy that not only ensures the retrieval of reliable advice but also maximizes the overall utility of the decision-making process (more details see Section 2.4).

1.3 Research Objectives

To address the challenges above, we propose a series of research objectives. Our research begins with binary decision-making, aiming to develop a multi-advisor binary decision-making method. Next, we employ this method in IRL scenarios. In addition, we consider the cost of querying advisors and the value of decisions to design an advisor selection and decision-making method with the objective of maximizing utility. Finally, we extend the binary decision-making method to make decisions among multiple options.

1.3.1 Sequential Binary Decision-Making without Ground Truth

To address the challenges presented in Section 1.2.1 and 1.2.2, our first objective is to design a Multi-Advisor Binary Sequential Decision-Making (MABSDM) method. Specifically, MABSDM involves making decisions on the binary decision-making problem, where multiple advisors make decisions on whether to agree or disagree with a given problem. For instance, bank staff needs to decide whether to lend money to borrowers, while advisors in investment agencies need to determine whether to invest in a project. To address the challenges mentioned in Section 1.2.1, our specific aim is to aggregate the advice provided by the multiple advisors in order to make decisions. Therefore, our first objective is:

O1.1 Designing a method for aggregating advice of advisors to make the optimal decisions for the binary decision-making problem.

In addition, following the challenges presented in Section 1.2.2, we aim to design a trustworthiness modeling method. Therefore, the second objective is:

O1.2 Proposing a trustworthiness modeling strategy that can model the advisors' trustworthiness over time.

1.3.2 Multi-Advisor Interactive Reinforcement Learning

Based on the challenges described in Section 1.2.3, our second objective is to design a Multi-Advisor Interactive Reinforcement Learning (MAIRL) system. The multi-advisor IRL has five similar characteristics to the multi-advisor sequential decision-making problem. First, MAIRL needs to aggregate the advice of multiple advisors for providing human rewards (decision-making of rewards given to the agent). Second, advisors often cannot access the ground truth because of reward sparsity. Third, we often cannot obtain prior information on advisors' trustworthiness, so we also need to build advisors' trustworthiness over time. Fourth, IRL is a time series problem. The advisors need to determine the rewards given to the agent at each time step, so it is also a sequential decision-making problem. Lastly, we use binary feedback as the feedback form in our IRL system. Therefore, we apply MABSDM (see Section 1.3.1) as a part of MAIRL in this objective. However, in addition to the problems that MABSDM can solve, we can still come up with some additional objectives to solve problems that exist independently in IRL.

First, in MAIRL, we adopt a binary feedback system, where human advisors provide two feedback signals, namely "good" or "bad" (Griffith et al., 2013; Arakawa et al., 2018). This binary feedback setup aligns with the binary decision-making scenario discussed earlier. However, before employing binary feedback, it is essential to evaluate its performance. Therefore, our first objective is:

O2.1 Evaluating the performance of binary feedback.

Second, we use MABSDM to aggregate the advice of multiple advisors to infer the correct rewards and model the trustworthiness of advisors over time.

Third, we aim to design a review mechanism within the MAIRL system to correct incorrect rewards. In more detail, human feedback is often expensive and time-consuming. In an IRL system, the designer often considers reducing the pressure on human advisors as much as possible. One way to reduce the feedback burden is to avoid giving rewards to the state-action pair that has been feedbacked before.

In the MAIRL system, the system records the feedback from advisors for each state-action pair. When the agent encounters the same state again, it retrieves the reward from memory, significantly reducing the need for human feedback. However, this approach introduces a challenge wherein future rewards are wrong if the previous reward is incorrect. Therefore, the MAIRL system requires a review mechanism to correct the inaccurate reward by re-evaluating those state-action pairs that have unreliable rewards. Thus, our second objective is:

O2.2 Designing a review model to correct the unreliable reward.

1.3.3 Sequential Binary Decision-Making by Maximizing the Utility

To address the challenge in Section 1.2.3, we make the third objective: designing a Multi-Advisor Dynamic Decision-Making (MADDM) system by maximizing the Utility.

First, we design a strategy for advisor selection based on the problem value and advisor cost. The first objective is:

O3.1 Designing an advisor selection model that can balance the cost and trustworthiness of advisors and the value of decisions.

Second, to complete MADDM, we combine the advisor selection model with MABSDM (see Section 1.3.1). Furthermore, to enhance the utilization of advisors' advice, a review model is introduced in MADDM. This model calibrates the trustworthiness of advisors by incorporating more accurate evidence obtained from reviewing historical problems based on the current advisors' trustworthiness. The second objective is:

O3.2 Designing a review model that can calibrate advisors' trustworthiness by reviewing historical problems.

1.3.4 Sequential Decision-Making among Multiple Options

To further address the challenges in Section 1.2.1 and 1.2.2, the last objective is to design a Multi-Advisor Sequential Decision-Making (MASDM) system to make sequential decisions among multiple options.

Specifically, following the challenge in Section 1.2.1, we aim to design a method that can aggregate the advice of multiple advisors. So the first objective is:

O4.1 Designing an aggregating method that can aggregate the advice of advisors to make optimal decisions among multiple options.

Following the challenges presented by Section 1.2.2, we aim to design a trustworthiness modeling method for the problems with multiple options, thereby the second objective is:

O4.2 Designing a trustworthiness modeling strategy that can model the advisors' trustworthiness over time for making decisions among multiple options.

1.4 Contributions

The contributions of this thesis are introduced as follows:

- To develop a sequential binary decision-making method (Objective **O1.1**, **O1.2** in Section 1.3.1), we propose the MABSDM method. To be specific, our approach considers (1) modeling the advisors' trustworthiness sequentially without prior information and ground truth, (2) automatically adjusting the aggregation strategy according to the uncertainty of the advisors' trustworthiness, and (3) making optimal decisions from the advice of multiple imperfect advisors. We evaluate our algorithm through several simulated experiments. The results show that our MABSDM has a better performance than three other methods that combine state-of-the-art models.

  This contribution is described in the following paper [(Guo et al., 2022)]:

  Zhaori Guo, Timothy J. Norman, Enrico H. Gerding, MTIRL: Multi-Trainer Interactive Reinforcement Learning System, ***PRIMA 2022: Principles and Practice of Multi-Agent Systems***

- To design a multi-advisor IRL system (Objective **O2.1**, **O2.2** in Section 1.3.2), we propose the MAIRL system, which can aggregate binary feedback from multiple imperfect advisors into a reliable reward for agent training in a reward-sparse environment. In addition, we employ binary feedback as the feedback form to minimize feedback pressure on advisors. Moreover, the review model in MAIRL can correct the unreliable reward advisors have given before. In particular, our experiments for evaluating feedback forms show that binary feedback outperforms other feedback forms including ranking feedback, scaling feedback, and state value feedback. Finally, we conduct grid-world experiments to show that the policy trained by the MAIRL with the review model is closer to the optimal policy than that without a review model.

  This contribution is described in the following paper [(Guo et al., 2022)]:

  Zhaori Guo, Timothy J. Norman, Enrico H. Gerding, Multi-Trainer Binary Feedback Interactive Reinforcement Learning System, ***Annals of Mathematics and Artificial Intelligence (under review)***

- To complete Objective **O3.1** and **O3.2** in Section 1.3.3 (developing a method for sequential binary decision-making by maximizing the utility), we propose MADDM, which is a novel strategy for optimally selecting a set of advisers in a sequential binary decision-making setting, where multiple decisions need to be made over time. Specifically, based on the MABSDM method, the MADDM method considers how to select advisors by balancing the advisors' costs, advisors' trustworthiness, and the value of making correct decisions. We evaluate MADDM through several numerical experiments. The results show that our approach outperforms two other methods that combine state-of-the-art models.

This contribution is described in the paper [(Guo et al., 2023a)]:

Zhaori Guo, Timothy J. Norman, Enrico H. Gerding, Multi-Trainer Binary Feedback Interactive Reinforcement Learning System, ***Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023)***

These three contributions above also are described in paper [(Guo et al., 2023b)]:

Zhaori Guo, Timothy J. Norman, Enrico H. Gerding, Multi-Advisor Dynamic Decision-Making, ***Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023, Doctoral Consortium)***

- Following Objective **O4.1**, **O4.2** in Section 1.3.4 (designing a method for the decisions among multiple options), we present a novel method MASDM that extends and improves MABSDM, to further address the challenges in Section 1.2.1 and 1.2.2. Compared to MABSDM, MASDM can make decisions among multiple options, not just binary options. To evaluate the performance of MASDM and its models, we conduct extensive experiments using the MNIST dataset (LeCun et al., 2010) and simulated data under different conditions. Specifically, we compare MASDM to the methods that combine state-of-the-art approaches, including the weighted voting method, the Bayesian method, and Beta distribution. The results show that MASDM outperforms the other methods in almost all simulated experiments and in most MNIST dataset experiments.

  This contribution is described in the paper [(Guo et al., 2023a)]:

  Zhaori Guo, Timothy J. Norman, Enrico H. Gerding, Multi-Advisor Sequential Decision-Making without Ground Truth, ***Journal of Autonomous Agents and Multiagent Systems (under review)***

1.5 Thesis Structure

The remainder of the thesis is structured as follows:

- Chapter 2 introduces the literature review and the limitations.
- Chapter 3 analyses our MABSDM method and its performance evaluation.
- Chapter 4 details our MAIRL method and studies its performance in grid-world experiments.
- Chapter 5 studies our MADDM method and its performance evaluation.

- Chapter 6 presents our MASDM method. The performance of these algorithms is tested by real data and simulated data experiments.
- Chapter 7 summarizes the thesis.

Chapter 2

Background and Literature Review

In this chapter, we present the background and literature review of our research. The chapter is structured as follows. First, in Section 2.1, we discuss advice aggregation in decision-making. Next, we proceed to detail trustworthiness modeling methods in Section 2.2. In addition, we focus on IRL in Section 2.3. Then, we introduce advisor selection methods in Section 2.4. Finally, we summarize the gaps and limitations in the present literature.

2.1 Advice Aggregation in Decision-Making

In this section, we focus on the challenge in Section 1.2.1 (making optimal decisions sequentially without ground truth), discussing research on aggregating advice of multiple advisors for decision-making. Specifically, the aggregating method is a way of

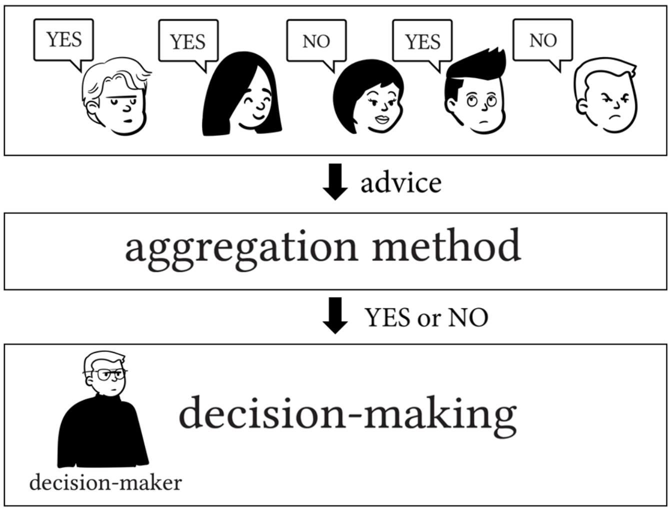


Figure 2.1 This is an example of the aggregation method for decision-making, the aggregation method needs to output a decision based on the advice of five advisors.

decision-making by aggregating the advice of advisors. For example, in Figure 2.1, there are five advisors who provide advice on a problem. But three of them think ”yes” is the correct decision, while others believe ”no” is correct. The aggregation method is required to aggregate their advice and give a final decision. Therefore, the input of the aggregation method is the advice of five advisors (can have more information, e.g., trustworthiness), while the output is the decision. In this section, we mainly introduce three commonly used aggregation methods, which are voting, value average, and Bayesian aggregation.

2.1.1 Voting

In the field of aggregating opinions from multiple advisors, voting is one of the most commonly used methods (Zhou, 2012; Dong et al., 2020). Specifically, the majority voting method is the simplest voting method, which uses the option with the most votes as the decision (Zhou, 2012; Kuncheva, 2014). For example, the idea of voting is used in the aggregation steps in *Random Forests* (Breiman, 2001). However, the majority voting method assumes that all the advisors have the same trustworthiness, which is often unrealistic in practice (Zheng et al., 2017; Chen et al., 2022b).

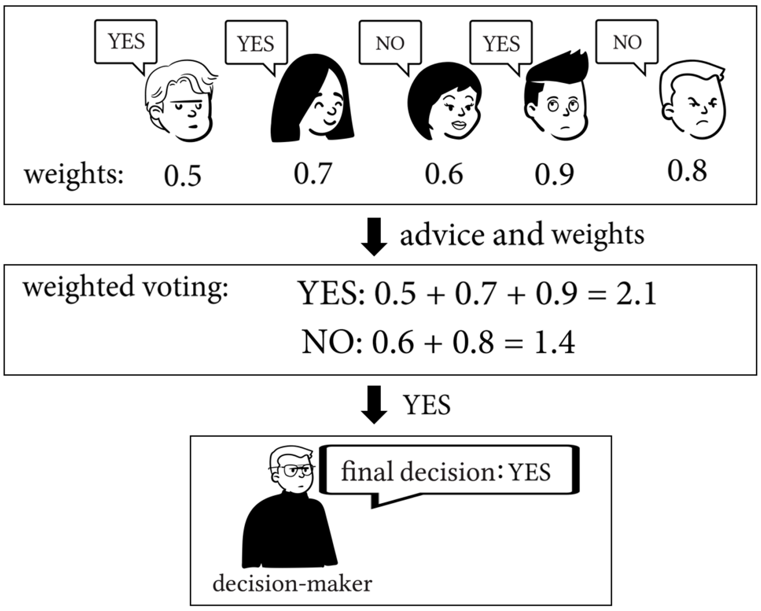


Figure 2.2 In this example, each advisor has a weight, which reflects his or her importance in decision-making. For the weighted voting method, the value of option ”yes” is bigger than the value of option ”no”. Therefore, the final decision is ”yes”.

Considering the limitation of the majority voting method, an improved method, *Weighted Voting* is often used. For instance, in the decision-making process, weighted voting considers the weight of advisors, which reflects the impact of the advice of the advisor on decision-making. Normally, a bigger weight of the advisor indicates more important advice, and the option with the largest cumulative voting value is the decision

(Kuncheva, 2014). For example, in Figure 2.2, there are five advisors with different weights and they provide different advice for a problem. If we utilize the weighted voting method for decision-making. We can obtain the value of the option ′′yes″ is 2.1, which is bigger than 1.4, the value of the option ″no″. Therefore, the decision is the option ″yes″.

There is considerable work related to the weighted voting method. For example, the ensemble machine learning methods, *AdaBoost* and *Gradient Boosting*, use weighted voting for aggregating the advice of multiple classifiers (Freund et al., 1996; Friedman, 2001). In addition, Davani et al. (2022), Liu et al. (2021), and Chen et al. (2022b) employ weighted voting to enhance the accuracy of decisions by modeling the trustworthiness of advisors to obtain more accurate aggregated advice. Moreover, Krawczyk et al. (2017) consider online aggregation by weighted voting. They continuously adjust the weights assigned to advisors to improve the aggregating performance through the data stream.

However, as we will show in this thesis (see more details in Chapter 3), weighted voting has low aggregating accuracy when each option is relatively independent and the reliability of trustworthiness is high. In contrast, our work does not always rely on WV, but instead uses it only when it is suitable, in order to maximize the aggregation performance.

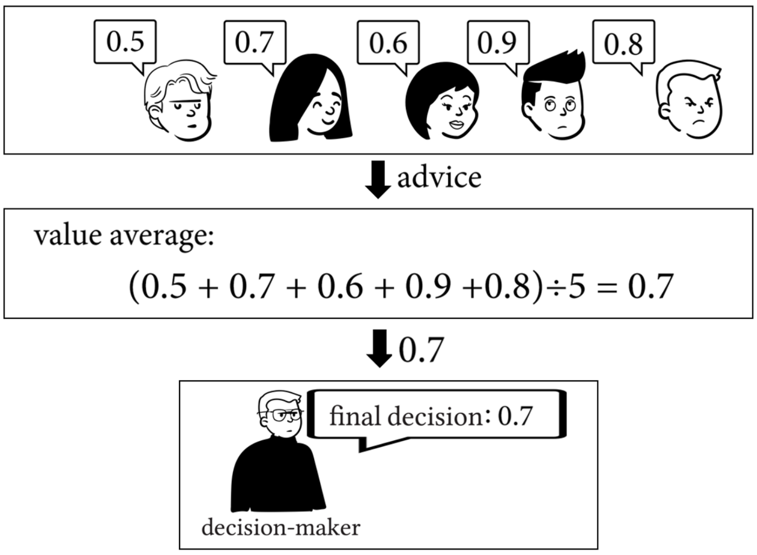


Figure 2.3 In the example of the value average method, the advice of five advisors is number or value. The output of the value average method is calculated from the average value of the advice of five advisors.

2.1.2 Value Average

Another aggregation method often used in decision-making is averaging. As the name suggests, averaging involves taking the average value of answers to arrive at the decision (Zhou, 2012; Perrone, 1993). Figure 2.3 provides an example of the value average method. In more detail, five advisors provide a number or value as their individual advice. Then, the decision made by the value average method is the average value of their advice, i.e., the decision is 0.7.

For the work related to the value average method, in ensemble machine learning, *Bagging* methods such as *Random Forest* use the averaging method for aggregation to predict the ground truth for regression problems (Breiman, 2001). In finance, averaging methods can be applied to aggregate the advice for investment decisions. For example, the mean-variance model in portfolio theory determines the optimal portfolio by calculating a weighted average of the expected returns of different assets (Markowitz, 1952). Furthermore, in consensus decision-making, the decision is often obtained by aggregating individual advice using the average method (Zhong et al., 2022).

However, the average method is primarily for decisions with continuous options, such as regression problems, while it is not reliable when there is no relationship between candidate options. For example, in a classification decision where advisors need to identify an animal in a picture, with options being cat, dog, squirrel, and sheep, the averaging method cannot be directly applied to average these categorical options. Therefore, if the categorical variables, like animal types, do not have a natural numerical scale or continuous relationship that allows for straightforward averaging, the averaging method cannot work effectively.

2.1.3 Bayesian Aggregation

Another commonly used aggregation method for decision-making is the Bayesian aggregation method, which uses the advice set and related information, e.g., advisors' trustworthiness, to calculate the correct probability of each option, and then the option with the highest probability is the decision (Kittler et al., 1998; Zhou, 2012; Kuncheva, 2014). In Figure 2.4, the Bayesian aggregation method calculates the correct probability (posterior probability) over all options according to the advice of advisors, then the option with the biggest probability is the decision.

The Bayesian aggregation method has been extensively studied and applied in decision-making processes. For example, some works (Dawid and Skene, 1979; Demartini et al., 2012; Karger et al., 2014; Sabetpour et al., 2021) estimate the correct decisions by the maximum likelihood estimation and improve the decisions and advisors' trustworthiness by *Expectation Maximization* (EM) method. However, the Bayesian aggregation

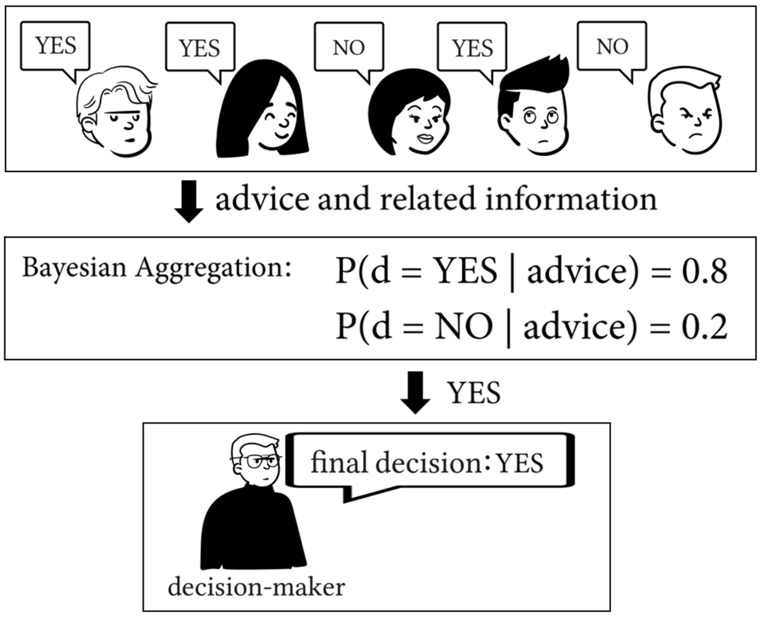

YES
YES
NO
YES
NO
advice and related information
Bayesian Aggregation: P(d = YES | advice) = 0.8
P(d = NO | advice) = 0.2
YES
final decision: YES
decision-maker

Figure 2.4 The Bayesian aggregation method utilizes the prior information, e.g., the advice of advisors, to compute the correct (posterior) probability over all options. In this example, the probability of the option "yes" is bigger than "no", so the decision is "yes".

method is easily influenced by multiple factors such as the difficulty of problems, the distribution of the ground truth of the options, and the correlation of advisors. Therefore, other works (Whitehill et al., 2009; Kim and Ghahramani, 2012; Venanzi et al., 2014; Li et al., 2019; Wu et al., 2023) consider such factors in the EM method to improve the decision accuracy. Although the Bayesian aggregation method works well, it relies heavily on prior knowledge of an advisor's trustworthiness. If this is not available or is unreliable, the Bayesian aggregation method may not work well (Ly et al., 2017).

Compared with previous works, our work is more challenging in we build the advisors' trustworthiness from zero without prior information. Therefore, in the initialization phase of the trustworthiness model, advisors' trustworthiness is unreliable, so the Bayesian aggregation method is not suitable for decision-making (see more details in Chapter 3).

2.2 Trustworthiness Models

Based on the challenge in Section 1.2.2 (modeling advisors' trustworthiness without prior information), in this section, we introduce two commonly-used approaches for modeling trustworthiness, which aim to quantify the trustworthiness of advisors. These two methods are the Beta distribution methods and matrix trustworthiness methods.

2.2.1 Beta Distribution

Beta Distribution is one of the most widely employed methods for modeling trustworthiness (Demartini et al., 2012; Karger et al., 2011; Liu, 2012; Fang et al., 2016; Liu et al., 2012). Beta Distribution expresses the advisor's trustworthiness as a probability between 0 and 1, using two values α and β to represent the evidence of correctly and incorrectly made decisions. The trustworthiness of the advisor can be expressed as $\alpha/(\alpha+\beta)$ (Johnson et al., 1995). Figure 2.5 provides an example of the trustworthiness

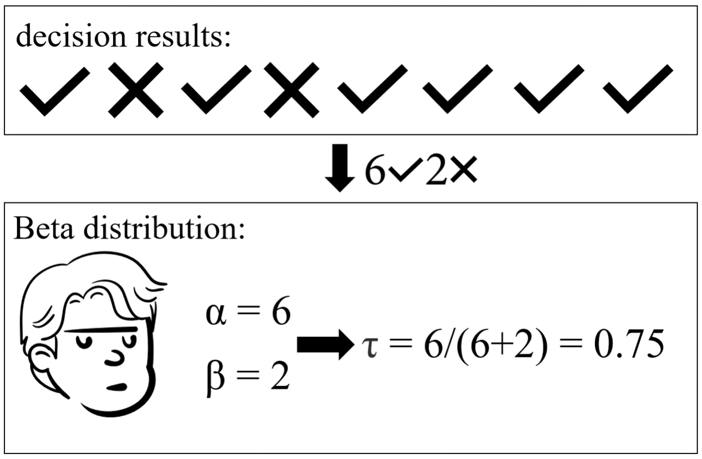


Figure 2.5 The Beta distribution method utilizes the number of times an advisor makes correct and incorrect decisions to model trustworthiness. In this example, the advisor provides 8 times of advice, which has 6 correct ones and 2 incorrect ones. Therefore, α is 6 and β is 2. Then, we can obtain the trustworthiness of the advisor is 0.75.

modeling by Beta distribution. In more detail, there is an advisor who makes 8 decisions. Among these decisions, 6 of them are right and 2 are wrong. Then we can obtain his trustworthiness is 0.75. The higher the trustworthiness is, the advisor has a higher probability of making a correct decision.

In addition, a method developed from the Beta distribution named *Subjective Logic* is proposed (Jøsang, 2016). This method is a trust modeling method that is widely used in artificial intelligence and decision theory (Burnett et al., 2010; Şensoy et al., 2013; Cerutti et al., 2015; Güneş et al., 2017; Cheng et al., 2021). Compared to the Beta Distribution, the advantage of Subjective Logic is that it directly quantifies the uncertainty of trustworthiness and specifies the initial trustworthiness of advisors to express intuitive beliefs (see more details in Section 3.1.2).

2.2.2 Matrix Trustworthiness

The matrix trustworthiness method is another method for trustworthiness modeling (Dawid and Skene, 1979; Kim and Ghahramani, 2012; Venanzi et al., 2014; Cui et al., 2021a; Wu et al., 2023). Specifically, in problems with n fixed options, the shape of the matrix is $n \times n$. For example, in binary decision-making problems, there is a matrix of advisor x $\tau_x = \begin{bmatrix} 0.1 & 0.9 \\ 0.7 & 0.3 \end{bmatrix}$. The value 0.9 (row 1 and column 2) indicates that he or she has 90% probability to choose the second option when the ground truth is the first option.

The trustworthiness model based on the matrix often performs better than those based on a single value when the distribution of the ground truth is unbalanced. However, it needs more evidence to support it than a single value. Its advantage cannot be guaranteed when the ground truth is relatively evenly distributed. Therefore, our work employs a single value to model trustworthiness because we focus on the general decision-making problem instead of that with special conditions.

2.3 Interactive Reinforcement Learning

Following the challenge in Section 1.2.3 (imperfect advisors in IRL), we discuss the literature about IRL in this section. IRL, also known as human-in-the-loop RL, is an approach that combines RL algorithms with human expertise to solve complex tasks. In this paradigm, humans participate in the learning process by providing guidance to an AI agent. One of the key benefits of IRL is its ability to reduce training time. Specifically, instead of relying solely on trial-and-error exploration, the agent can leverage human guidance to focus its exploration efforts on the most promising areas of the task space. This interaction can lead to more efficient learning and faster convergence toward optimal solutions (Knox and Stone, 2008a).

The interaction between humans and AI agents in IRL can take various forms. For example, humans can provide explicit rewards or penalties to guide the agent's behavior. In addition, the human can also serve as an advisor, giving the agent action advice when it is in a tricky state.

In this section, we describe the background and related work of IRL. Specifically, we introduce the background of RL in Section 2.3.1. Section 2.3.2 discusses the methods of reward shaping. Next, Section 2.3.3 proceeds to detail the work of advice-based IRL methods. Moreover, we turn our attention to the work related to the feedback forms in Section 2.3.4. Finally, we discuss the work related to multiple resource IRL in Section 2.3.5.

2.3.1 Background of Reinforcement Learning

RL is a paradigm of machine learning where an agent learns optimal behavior by interacting with an environment and receiving feedback in the form of rewards or penalties. The primary objective is to find a policy that maximizes the expected cumulative reward over time. The foundational elements of a RL problem include:

- **Agent**: The entity that observes, learns, and makes decisions.
- **Environment**: The external context with which the agent interacts.
- **State**: A representation of the environment.
- **Action**: Decisions the agent can make.
- **Reward**: Immediate feedback from the environment, indicating the value of the action taken.
- **Policy**: A mapping from states to actions, defining the agent's behavior.
- **Value Function**: A function estimating the expected cumulative reward from a given state or state-action pair.

The RL process can be visualized as a loop of interactions between the agent and the environment. The typical steps are:

1. **Initialization**: Initialize the policy and the value function.
2. **Observation**: The agent observes the current state s_t of the environment.
3. **Decision**: Based on the policy and current state s_t, the agent selects an action a_t.
4. **Interaction**: The agent performs the action a_t in the environment.
5. **Feedback**: The environment transitions to a new state s_{t+1} and provides a reward r_t to the agent.
6. **Learning**: The agent updates its policy and/or value function based on the observed transition (s_t, a_t, r_t, s_{t+1}).
7. **Loop**: The agent then uses state s_{t+1} as the new current state and repeats the process.
8. **Termination**: The process continues until a termination condition is met, such as reaching a maximum number of steps, achieving a satisfactory level of learning, or encountering a terminal state in the environment.

2.3.2 Reward Shaping Method

Reward shaping refers to the process of modifying the reward function to guide the learning agent toward desired behaviors and expedite the learning process. It adds additional reward components or adjusts existing ones to provide more reward signals to the agent. Based on the composition of rewards, the reward shaping method can be classified into two categories: only human reward and integrating reward. We will discuss the work related to these two methods in the following sections.

2.3.2.1 Only Human Reward

As early as 1998, Stern et al. (1998) discover that similar to animal trainers, virtual pets can learn human preferences through rewards and interact in a friendly manner with humans. Similarly, Isbell et al. (2001) apply human rewards to social dialogue robots based on RL technology. By continuously providing rewards, people are able to correct the robot's policy, leading to the agent gradually understanding human preferences after more than 3,000 reward instances. In addition, Thomaz et al. (2005) extend this idea and applies the reward shaping method to the robot. Specifically, they not only consider human rewards but also introduce action guidance, allowing humans to correct the robot's past behavior and guide the agent's future actions.

However, previous works often require a high knowledge level of human trainer (Knox and Stone, 2008a). To address this problem, Knox and Stone (2008a) improve the way of human reward and proposes the *Training an Agent Manually via Evaluative Reinforcement* (TAMER) framework. Specifically, instead of requiring professional advice, humans are only asked to provide scaling rewards (i.e., rewards from a range of values, e.g., from 1 to 100). This setting makes the task training process more effective. They conducted experiments using the Tetris game and demonstrated that, with just three rounds of rewards, the agent is able to achieve good performance in the game. This approach significantly reduces the burden on non-expert participants and enhances the accessibility of RL training. Furthermore, to enhance the sample efficiency of human reward, Lee et al. (2021) propose a pre-training approach that updates the reward function based on human preferences. This method is particularly applicable in complex tasks, such as mechanical manipulation skills, where the agent can acquire new behaviors that standard reward functions alone may not effectively capture.

In addition, Warnell et al. (2018) employ convolutional neural networks to model a value network and utilize human rewards to update the network's parameters. This approach enables the application of IRL in high-dimensional state spaces. Their results show that the agent trained for only 15 minutes outperforms human players in the Atari bowling game. Similarly, da Cruz et al. (2021) leverage human rewards to train a Deep Q Network for labeling breast disease images. Furthermore, reducing the number of human queries is an effective strategy for alleviating feedback pressure. For example, Bignold et al. (2021b) categorize different states based on distinct rules, assigning similar rewards to states governed by the same rules. This method significantly reduces the amount of required human rewards.

However, relying solely on human rewards can lead to good short-term performance for the agent, but the long-term performance is not ideal because it is challenging for human rewards to cover all situations with limited feedback instances (Knox and Stone, 2015). Moreover, due to the substantial uncertainty and instability of human rewards, if the accuracy of human rewards cannot be guaranteed, this method is likely to hinder the agent in finding the best policy (Ng et al., 1999).

2.3.2.2 Integrating Reward

In IRL, the integrating reward method refers to the agent training by the rewards from the environmental rewards and the human rewards. By integrating these rewards, the agent can learn the policy effectively and take actions that align with both the environmental objectives and the preferences expressed by human experts.

A pioneer work, Thomaz and Breazeal (2008) explore the compatibility of human reward and environmental reward using RL robots. They find that human reward signals play a crucial role in guiding future action choices. In addition, Tenorio-Gonzalez et al. (2010) incorporate human feedback signals as additional rewards. Specifically, depending on the agent's performance, their method allows humans to provide further rewards, which are used as additional rewards to help the agent's learning. Moreover, the TAMER + RL method is a new attempt based on the TAMER framework (Knox and Stone, 2010). In this method, the authors incorporate human reward into the feature vector of reward to enhance the learning process. Subsequently, Knox and Stone (2015) improve this combination and proposed *Value Iteration TAMER* (VI-TAMER), which combines human binary feedback with a value function and uses the Temporal Difference method to update the policy. Their results show that VI-TAMER has a better performance in the mountain car and grid-world games than the benchmarks. In more detail, VI-TAMER, especially in continuous tasks, consistently reaches the goal more effectively compared to other algorithms. In addition, the agent using VI-TAMER reaches the goal faster during the initial stages of training compared to the benchmarks.

Other works focus on giving reward signals through different interactive ways. For example, Arakawa et al. (2018) propose the *Deep Q Network TAMER* (DQN-TAMER) method. In more detail, they employed facial expressions as the reward signals, converting them into binary signals within the neural network processor. The agent is then trained by combining human and environmental rewards. Their results show that DQN-TAMER outperforms the benchmark in a maze game scenario. Similarly, Li et al.

(2020) also utilize facial expressions as the feedback way. They claim that facial reward signals can improve training efficiency through a study involving 561 participants.

Furthermore, based on DQN-TAMER, Guan et al. (2020) employ a deep neural network to model the policy function and extends the facial rewards to include voice rewards. In addition, another work related to voice feedback is Jeon et al. (2021), which uses automatic speech recognition to convert human commands into digital rewards for training DQN. Moreover, Cui et al. (2021b) propose the general implicit rewards framework that can map various forms of implicit human signals to the rewards for agent learning.

Overall, the integrating reward method can provide long-term effects on agent training and can accelerate the learning process (Amir et al., 2016; Li et al., 2013; Fachantidis et al., 2019). Even when humans no longer provide rewards after a certain period, the agent can continue to learn the optimal policy solely based on environmental rewards. However, determining appropriate human rewards remains a complex problem, and a definitive criterion for setting them has yet to be established. When designing human rewards, it is crucial to consider their consistency with environmental rewards in order to ensure effective learning.

2.3.3 Action Advice Method

The action advice method involves humans providing advice on the next action for the agent based on its current state. In contrast to the reward shaping method influencing the policy of the agent, human action advice is often independent of the policy and it does not require modifying the environmental reward structure. Specifically, in the action advice method, humans indicate the optimal action to the agent, while the agent still receives rewards from the environment. For example, Suay and Chernova (2011) apply the action advice method to guide real robots in performing tasks. In more detail, they evaluated the performance of the action advice method in state spaces of varying sizes and found that this interaction method performed better in larger state space environments than in smaller environments.

In addition, Krening and Feigh (2019) introduce the Newtonian action advice method, which combines human advice with RL techniques. Specifically, this method facilitates agent learning by incorporating human action suggestions at regular intervals. In addition, a distinctive feature of this approach is the decay of human-provided action suggestions over time, ensuring their effectiveness for a certain duration. This mechanism reduces the cognitive burden on humans while maintaining the learning efficiency of the agent. Their results demonstrated that the Newtonian action advice method outperforms IRL methods related to the reward shaping method. Furthermore, Krening and Feigh (2018) compare the Newtonian action advice method with the binary feedback method, and the results indicate that the action advice method contributed to more efficient agent learning than the binary feedback method. Moreover, Frazier and Riedl (2019) combine the Newtonian action advice method with a deep RL algorithm and applies it in the Minecraft game. The experimental results demonstrate that this combined approach outperforms the deep RL method alone.

In addressing the costs and limitations associated with human feedback, Bignold et al. (2021a) adopt simulated users with explicit rules to assist the agent in improving learning efficiency instead of using clear action advice. Although the performance of simulated users may not match that of real humans, they provide immediate advice to the agent without any delay, thereby mitigating the drawbacks of human feedback. Moreover, the action advice method can also be applied in the domain of safe IRL, where humans assist the agent in avoiding dangerous actions by providing advice on the safest actions to take (Li et al., 2022).

However, the action advice method gives clear instructions to the agent, but it is provided often before the agent performs an action. Compared to the reward shaping method, action advice is more likely to provide wrong guidance for the agent when human trainers cannot predict the state after the agent executes an action (Arzate Cruz and Igarashi, 2020).

2.3.4 Human Feedback Forms

There is no consensus on the best form of feedback to use in the IRL community. Specifically, some studies employed binary feedback (Isbell et al., 2001; Arakawa et al., 2018; Chisari et al., 2022), which needs human trainers to provide binary rewards for agent training. The advantage of binary feedback is that it requires less feedback burden than other forms. Other studies (Knox and Stone, 2008b; MacGlashan et al., 2017) adopt the scaling reward, where the reward signal is a score within a specified range. Furthermore, the works (Fachantidis et al., 2019; Frazier and Riedl, 2019; Chen et al., 2022a) have considered action advice methods, where humans need to provide the agent with action advice rather than a reward.

However, there is no work that analyzes the impact of different feedback forms on the training of RL agents, which highlights the need for further research to determine the best feedback form in IRL. Specifically, a deeper understanding of the impact of different forms of reward on the training process and the resulting performance of the RL agent is crucial for the continued advancement of this field. In this thesis, we conduct experiments to evaluate different feedback forms (see Section 4.2.2).

2.3.5 Multiple Resource Interactive Reinforcement Learning

Some works (Zhan et al., 2016; Li and Zhang, 2018; Kurenkov et al., 2020) in the field of IRL have enabled multiple trainers to participate in the training process. For example, Li and Zhang (2018) focuses on source policy selection in transfer RL. Specifically, they consider selecting source policies as a multi-armed bandit problem and employ the *Upper Confidence Bound* method to solve it. Moreover, Kurenkov et al. (2020) propose a filtering approach that compares Q-values to filter out bad advice from trainers.

However, these methods only select one trainer at a time, which can result in poor training accuracy if the feedback provided by the selected trainer is inaccurate. In addition, the approaches rely on the availability of environmental rewards as ground truth, which may not be feasible when the rewards are highly sparse. In contrast, our method does not rely on ground truth and can effectively utilize all feedback from multiple trainers and produce more accurate and cost-effective training results.

2.4 Introduction to Multi-Armed Bandit Problems

Following the Challenge in Section 1.2.4, we discuss the existing approaches and the works related to the multi-armed bandit problem of advisor selection.

2.4.1 Existing Approaches for Multi-Armed Bandit Problem

In Chapter 5, in our methods and experiments, we use three methods for solving the multi-armed bandit problem, they are: *ϵ-greedy*, *Thompson Sampling*, and *Upper Confidence Bound Algorithm* (UCB). Therefore, we discuss them in the following.

2.4.1.1 ϵ-Greedy

ϵ-greedy (Sutton and Barto, 2018) is a widely employed algorithm used to solve the multi-armed bandit problem. It is a simple and intuitive approach that strikes a balance between exploration and exploitation. In the ϵ-greedy method, the decision-maker maintains an estimate of the expected reward for each arm. At each time step, the decision-maker chooses between exploration and exploitation based on a parameter called ϵ. For example, if $\epsilon = 0.1$, the decision-maker selects a random arm with the probability of 10%, regardless of its estimated reward. This random exploration allows the decision-maker to gather more information about the arms and their reward distributions. It helps in identifying potentially better-performing arms that may be initially underestimated.

On the other hand, with a probability of $(1 - \epsilon)$, the decision-maker selects the arm with the highest estimated reward by exploiting its current knowledge, to maximize the decision-maker's immediate reward. By adjusting ϵ, the decision-maker can control the trade-off between exploration and exploitation. A higher ϵ encourages more exploration, while a lower ϵ prioritizes exploitation.

However, the ϵ-greedy method explores with a constant probability, regardless of the current estimates or the potential for improvement. Therefore, the ϵ-greedy method may continue to explore arms even after identifying high-reward arms. This can result in wasting cost and delay the exploitation of the best arm, especially when the ϵ value is set too high.

2.4.1.2 Thompson Sampling

Another extensively utilized method is Thompson Sampling (Thompson, 1933). It is a Bayesian approach that balances exploration and exploitation by using probability distributions to model uncertainty in the rewards of different arms. Specifically, the algorithm starts by assigning a prior distribution (e.g., Beta distribution) to each arm's reward probability. As the decision-maker selects arms and receives rewards, the algorithm updates these distributions based on the observed outcomes. Figure 2.6 provides

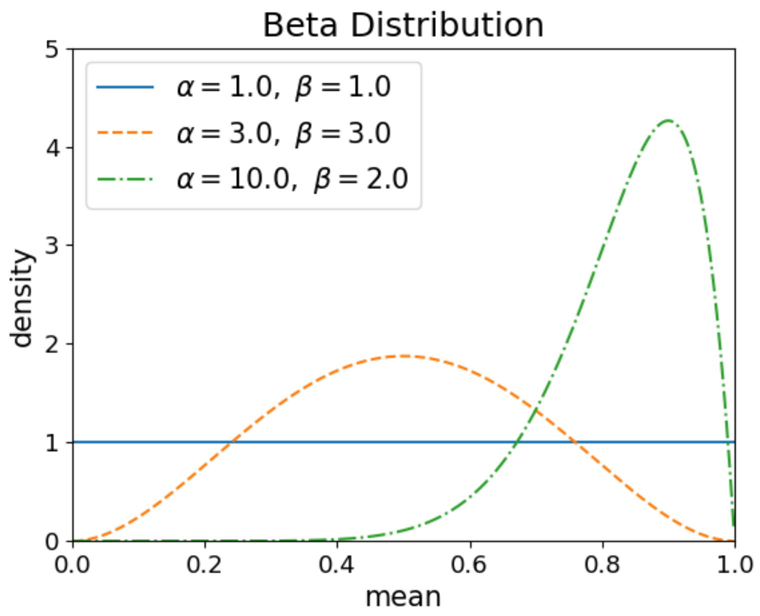


Figure 2.6 The x-axis represents the range of the mean or expected value of the Beta distribution. The y-axis represents the value of the probability density function of the Beta distribution at a given value of x.

an example of probability density images under three conditions in the Beta distribution. In the first condition, when $\alpha = 1$ and $\beta = 1$, the horizontal line represents that we do not have any prior information, so a value from the Thompson Sampling is sampled uniformly between 0 and 1. In addition, when $\alpha = 3$ and $\beta = 3$, the probability density image resembles a parabola with a peak at $x = 0.5$. In this case, the value from the Thompson Sampling likely is a value close to the mean value 0.5 (exploitation), but there is also a certain probability of obtaining some extreme values, e.g., 0.9 or 0.1 (exploration). Moreover, when $\alpha = 10$ and $\beta = 2$, the curve gradually converges and becomes narrower than the other two curves. The value from the Thompson Sampling is most likely around 0.85, and it is unlikely to get a value below 0.4.

Therefore, to select arms for decisions, Thompson Sampling leverages the posterior distributions. It samples a probability from each arm's posterior distribution and chooses the arm with the highest sampled value. This random sampling process naturally encourages the exploration of arms with uncertain rewards and the exploitation of arms with highly estimated rewards. By continuously updating the distributions and making decisions based on random samples, the posterior distributions of arms become more concentrated around the true probability, leading to more accurate decisions.

2.4.1.3 Upper Confidence Bound Algorithm

The UCB algorithm (Li et al., 2010) is another widely employed algorithm for solving the multi-armed bandit problem. It is designed to strike a balance between exploration and exploitation by leveraging the upper confidence bound to estimate the potential rewards of different arms. The key idea of UCB is to balance exploitation (utilizing known good arms) and exploration (trying unknown arms) by maintaining an upper confidence bound for each arm.

In the UCB algorithm, each arm is associated with an upper confidence bound, which represents the uncertainty in the estimated reward. The upper confidence bound is calculated based on the arm's observed rewards and the number of times it has been selected. Specifically, at each time t, for each arm o, maintain the average reward $\bar{r}_o$ and the number of times selected n_o. At each step, calculate the UCB for each arm o: $UCB_o = \bar{r}_o + \sqrt{\frac{2 \ln n}{n_o}}$, where $\bar{r_o}$ represents exploitation and $\sqrt{\frac{2 \ln n}{n_o}}$ represents exploration, and n is the total number of selections. Then, the decision d_t is the arm with maximum UCB, i.e., $d_t = \arg\max_o UCB_o$. Next, we receive the reward r_t of arm o, and update $\bar{r}_o$, n, and n_o. After that, we repeat the above steps at each time step.

Therefore, the selection of arms based on their upper confidence bounds ensures that the algorithm systematically explores different arms in the early stages to gather information and gradually shifts towards exploiting arms with higher estimated rewards as more data is collected.

2.4.2 Related Work of Multi-Armed Bandit Problem

Research grounded in multi-armed bandit methods is also relevant here (Kurenkov et al., 2020; Tran-Thanh et al., 2014, 2012; Xia et al., 2015). Specifically, Kurenkov et al. (2020) use Thompson Sampling to select advisors for agent training in RL. In their scenario, there is a set of imperfect advisors with different trustworthiness, and the agent needs to find the best advisor based on the advisors' advice over time. In addition, Tran-Thanh et al. (2012) propose an upper confidence method to solve budget–limited multi-armed bandit problems. However, these works assume that the ground truth is available following each decision, which means that advisors' trustworthiness can be reliably updated. Instead, our work infers the reliability of the advice by the decision model, thereby avoiding the need for ground truth. In more detail, assessing the reliability of the advice can help us give reasonable updated evidence for building models of the trustworthiness of advisors.

In addition, some studies consider advisor selection limited by a fixed budget constraint (Tran-Thanh et al., 2012, 2014; Xia et al., 2015; Zhou and Tomlin, 2018; Cayci et al., 2020). They often consider how to balance exploration and exploitation to maximize rewards in budget-constrained decision-making scenarios. However, such works do not consider that decisions might have different values and costs associated with getting them wrong. Consider the following lending decision scenario. If a $1,000 loan at 9% interest is repaid, it will make a $90 profit, but it can result in a loss of $1,000 if the borrower defaults. Such high-risk decisions require a more reliable assessment, potentially requiring multiple costly advisers, whereas low-value, low-risk decisions may only need a single one. Therefore, we consider selecting a group of advisors with different qualities and prices to balance potential profits and risks associated with a decision.

Moreover, the cost of advisors is considered in the work (Tong et al., 2018; Wang et al., 2018; Miao et al., 2022). In more detail, Tong et al. (2018) focus on pricing the advisors in different regions and deciding by the relationship between supply and demand in spatial crowdsourcing tasks. However, they do not consider advisors to have different qualities. In addition, Miao et al. (2022) and Wang et al. (2018) also assume advisors cost the same but give additional rewards to advisors with more contributions. However, when there is no real-time feedback on the ground truth, it is difficult to determine who should obtain additional rewards. Therefore, the same price for advisors with different qualities is unrealistic; in contrast, our work considers advisors with different qualities and prices.

2.5 Summary

This chapter presents the literature review relevant to our research objectives and challenges. We start by providing an overview of advice aggregation in the multi-advisor decision-making problem. This is associated with our research challenge in Section 1.2.1, making optimal decisions sequentially without ground truth. Specifically, although the aggregation problem has been widely studied in recent years, most of them fail to account for the impact of the changing trustworthiness, which is an important feature in sequential decision-making. In addition, existing research mainly focuses on offline methods, which cannot fit the dynamic nature of sequential decision-making. In contrast, our work considers designing an online method to aggregate advice sequentially to make optimal decisions, which can automatically adjust the aggregation strategy based on the change in advisors' trustworthiness (Objective **O1.1** and **O4.1**).

Next, we turn our attention to modeling advisors' trustworthiness without prior information (research challenge in Section 1.2.2). Specifically, existing methods for updating trustworthiness often rely on knowing the ground truth, while our work considers scenarios without having access to this information. In addition, although some work updates the trustworthiness without ground truth, they often depend on a big group of data (a large number of problems and advice from advisors) to estimate the trustworthiness. In contrast, our work considers gradually building advisors' trustworthiness from zero without ground truth and prior information (Objective **O1.2** and **O4.2**).

Then, we present the background and literature in the IRL, which is an implement area of multi-advisor decision-making problem (research challenge in Section 1.2.3). Specifically, existing research mainly focuses on single trainer IRL because they assume the trainer is perfect. However, the human trainer is hard to be perfect in practice, so we consider aggregating the advice of multiple trainers to provide the aggregating rewards for agent training. In addition, a few of the research consider multi-trainer IRL, but they often require that they can access the ground truth. In addition, our work also considers correcting unreliable historical advice to improve the reward accuracy (Objective **O2.2**). Furthermore, there is no consensus on the best form of reward to use in the IRL community. Therefore, our Objective **O2.1** focuses on evaluating the performance of different feedback forms.

Finally, we focus on a utility optimization problem in multi-advisor sequential decision-making. In this problem, we consider advisor selection by balancing the advisors' costs and the value of decisions (research challenge in Section 1.2.4). In more detail, although some works consider the budget limitation to control the cost of hiring advisors, they assume that all the decisions have the same value, while our work considers the advisor selection based on decisions that have various values (Objective **O3.1** and **O3.2**).

Chapter 3

Sequential Binary Decision-Making

In this chapter, we present a novel method, called Multi-Advisor Binary Sequential Decision-Making (MABSDM), for binary sequential decision-making problems, with the aim of addressing Objective **O1.1**, **O1.2** (see Section 1.3.1). MABSDM is the basis for subsequent extensions in the following two chapters. In more detail, MABSDM considers (1) modeling the advisors' trustworthiness sequentially without prior information and ground truth, (2) automatically adjusting the aggregation strategy according to the uncertainty of the advisors' trustworthiness and, (3) making optimal decisions by the advice of multiple advisors. Moreover, our problem-advice experiments show that our MABSDM method outperforms the benchmarks using state-of-the-art methods including the weighted voting method, the Bayesian aggregation method, and the Beta distribution trustworthiness model.

The rest of the chapter is structured as follows. First, we describe the MABSDM method in Section 3.1. Second, we present the experiments and results in Section 3.2. Lastly, we summarize our work in Section 3.3.

3.1 Model Description

MABSDM can aggregate the advice of multiple advisors to make the optimal decision for problems with binary options. The design of MABSDM considers two parts. The first part is the trustworthiness model. In more detail, the trustworthiness model can build the trustworthiness of advisors without ground truth over time. The second part is the decision model, which can make optimal decisions based on the advice of advisors and the present trustworthiness of advisors. In particular, the trustworthiness model and decision model are interdependent. For each problem, the decision model requires the present trustworthiness of advisors to make a decision, while the update of advisors' trustworthiness relies on the inferred evidence from the decision. Moreover, the two models in MABSDM improve over time without the ground truth of problems and prior information of advisors' trustworthiness. Figure 3.1 shows its structure.

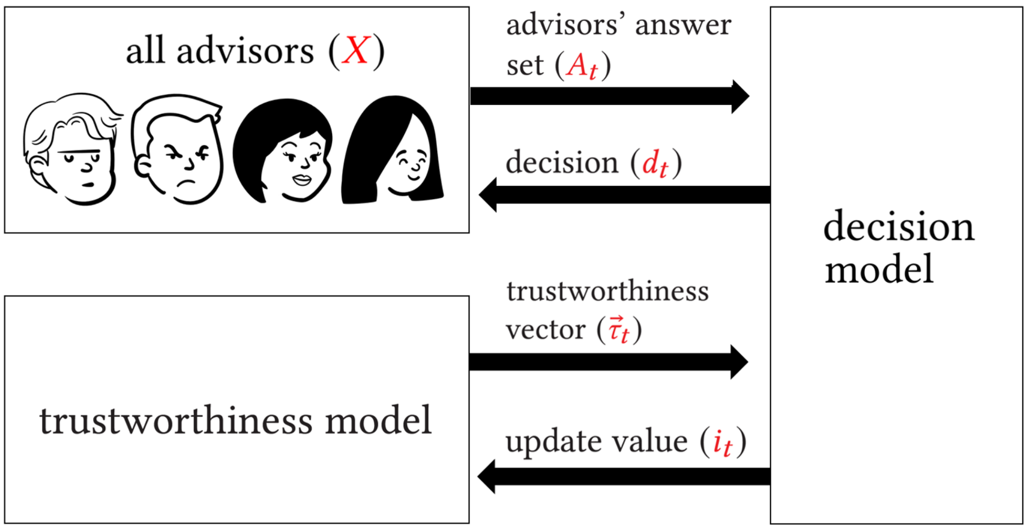


Figure 3.1 For each problem t, a subset of the advisor set X provides a set of advice A_t. The decision model takes two inputs to make the decision: the advice set A_t from advisors and the trustworthiness vector $\vec{\tau}_t$ from the trustworthiness model. Moreover, the decision model outputs the decision d_t and an update value vector $\vec{i}_t$. This value is then passed to the trustworthiness model to update the trustworthiness values of the advisors for the subsequent problem $t+1$.

3.1.1 Problem Formalization

The binary decision-making problem is a prevalent type of decision wherein advisors provide advice by choosing between two options. For instance, a bank decides whether to approve a loan application, and a factory determines whether to produce a specific commodity. More formally, the decision maker is faced with a sequence of problems $T = \{1, 2, \ldots, q\}$ for which it needs to make a decision. For each problem $t \in T$, the decision maker queries a set of advisors X. In sequential decision-making, problems appear, and a decision needs to be made, in order and without knowledge of future problems. Hence, we sometimes refer to t as a time step. In this chapter, we consider binary decision-making problems, and so the outcome of each problem is a decision $d_t \in \{pos, neg\}$ where pos is a positive decision (e.g. the loan is granted) and neg is a negative one. Although all advisors are asked for each question, not all advisors will respond, e.g., because they may not have appropriate expertise. Let $Y_t \subseteq X$ denote the set of advisors responding to t. Furthermore, let $A^{pos}_{Y_t} \subseteq Y_t$ denote the set of advisors that advise a pos decision. Similarly, let $A^{neg}_{Y_t} \subseteq Y_t$ denote the set of advisors that believe neg is the correct one. Let $A_t = \{A^{pos}_{Y_t}, A^{neg}_{Y_t}\}$ denote the advice set of problem t. For any $A^{pos}_{Y_t}$, $A^{neg}_{Y_t}$, we have $A^{pos}_{Y_t} \cap A^{neg}_{Y_t} = \emptyset$ and $A^{pos}_{Y_t} \cup A^{neg}_{Y_t} = Y_t$.

In addition, in cases where the advice of multiple advisors can conflict, decision-making considers the trustworthiness of advisors, which represents the degree to which an advisor's participation influences decision-making. After decision-making for each problem, trustworthiness is updated. More formally, for each problem $t \in T$, the decision d_t also depends on the trustworthiness of the advisors (see Section 3.1.3). For each advisor, $x \in X$, τ_t^x is the trustworthiness of x. Finally, we denote with $\vec{\tau}_t$ the vector containing all the advisors' trustworthiness values.

Moreover, we use $d_t = f(A_t, \vec{\tau}_t) \in \{pos, neg\}$ to refer to the decision-making function. For each problem t, let d_t^* denote the ground truth of the problem t, but this ground truth is never revealed to the advisors and decision-makers. If $d_t = d_t^*$, we consider that the decision is correct. Let ν denote the number of correct problems, i.e., $\nu = |\{t \in T | d_t^* = f(A_t, \vec{\tau}_t)\}|$. Our goal is to maximize the number ν by the decision-making function $f(\cdot)$.

3.1.2 Binary Cautious Trustworthiness Model

The *Binary Cautious Trustworthiness* (BCT) model is a trustworthiness model for building the trustworthiness of advisors over time. Specifically, trustworthiness is the degree to which the advisor is accurate in terms of providing advice that matches the ground truth. For instance, the advice provided by a highly trusted advisor holds greater significance compared to that of someone with lower trustworthiness. In addition, the challenge is that the BCT model needs to construct trustworthiness from zero without preliminary information and reliable evidence because the advisors do not have access to the ground truth. In this work, we utilize Subjective Logic and a cautious trustworthiness update strategy to construct our BCT model.

To begin with, we recall that Subjective Logic is a trustworthiness modeling method that is widely used in artificial intelligence and decision theory (see Section 2.2.1). Compared to the Beta distribution, Subjective Logic possesses two properties that align with our work: first, it can directly quantify the uncertainty of advisors' trustworthiness in the range $(0, 1]$. This uncertainty is used for building our decision model (see Section 3.1.3); second, we can specify the base rate (prior probability without any evidence) of advisors' trustworthiness to express cautious or intuitive beliefs because a cautious base rate can reduce the risk of wrong assessment of advisors' trustworthiness when there is no ground truth of problems and prior information of advisors' trustworthiness (see later in this section for more details). In addition, the update of advisors' trustworthiness often relies on the ground truth information. However, within our context, the lack of observable ground truth renders us unable to ascertain the correctness of decision d_t. If we update the trustworthiness in the wrong direction, incorrect updates could potentially propagate adverse effects on subsequent decisions. To solve this problem, we adopt a cautious strategy for updating advisors' trustworthiness (see later in this section for more details).

Specifically, for each advisor $x \in X$, for problem t, we define $b_t^x \in [0,1]$ as the belief in advisor x, and $e_t^x \in [0,1]$ as the disbelief in advisor x. In a decision-making scenario, $b_t^x \in [0,1]$ indicates the belief that the advisor x will make the correct decision, and it increases with the proportion of correct decisions. In contrast, $e_t^x \in [0,1]$ indicates the disbelief that the advisor x will make the correct decision. Moreover, we use $\theta_t^x \in (0,1]$ to denote the uncertainty of advisors' trustworthiness, which decreases as the evidence accumulates. The bigger θ_t^x reflects the lower reliability of the advisor's trustworthiness.

In addition, for each advisor x at problem t, we associate two strength parameters: α_t^x, β_t^x. Here, α_t^x is the evidence of advisor x that agrees with the decisions. In contrast, β_t^x is the evidence of advisor x that disagrees with the decisions. In this work, these two values are built through the accumulating evidence by sequential decision-making (see in Equation 3.4 and 3.5). Then, b_t^x, e_t^x, and θ_t^x can be expressed as:

$$b_t^x = \frac{\alpha_t^x}{\alpha_t^x + \beta_t^x + 2}, \quad e_t^x = \frac{\beta_t^x}{\alpha_t^x + \beta_t^x + 2}, \quad \theta_t^x = \frac{2}{\alpha_t^x + \beta_t^x + 2} \tag{3.1}$$

where the number 2 in Equation 3.1 represents the non-informative prior weight (Jøsang, 2016; Jøsang et al., 2022). Then we have $b_t^x + e_t^x + \theta_t^x = 1$. Moreover, let η_x denote the base rate of the trustworthiness without any evidence. The advisors' trustworthiness $\tau(x)$ can be expressed as:

$$\tau_t^x = b_t^x + \eta_x \theta_t^x \tag{3.2}$$

Furthermore, if the model does not have any prior knowledge about the distribution of the advisors' trustworthiness, BCT adopts a cautious strategy to set $\eta_x = 1/2$, which represents an equal probability for each advisor to choose any option. This approach ensures that BCT does not place excessive trust or mistrust in any advisor in the absence of any prior information. Of course, if there is reasonable prior knowledge available, an appropriate base rate or an intuitive base rate can be set accordingly.

Moreover, the BCT model employs a cautious strategy for trustworthiness updating, which is closely related to evidence α_t^x and β_t^x (the computation of α_t^x and β_t^x see Equation 3.4 and 3.5). Normally, the update of the trustworthiness relies on the ground truth. Taking a coin flip as an example, if we observe 4 heads and 6 tails in ten flips, then we can express α_t^x and β_t^x as $\alpha_{10}^x = 4$, $\beta_{10}^x = 6$. If the result of the 11th coin toss is a head, which indicates the new evidence is 1, then α increases by 1, i.e., $\alpha_{11}^x = 5$, and $\beta_{11}^x = 6$. However, there is no ground truth for obtaining the new evidence, so the BCT model employs a cautious strategy for trustworthiness updating to reduce incorrect updates. The cautious strategy updates trustworthiness based on the confidence of the decision. Specifically, our decision model outputs two probabilities $P_t^{pos} \in [0,1]$

and $P_t^{neg} \in [0,1]$, representing the advisors' beliefs in the correctness of the "positive" and "negative" advice, respectively (see more details in Section 3.1.3.3). Subsequently, a confidence value $i_t \in [0,1]$, derived from P_t^{pos} and P_t^{neg}, can be expressed as follows:

$$i_t = |P_t^{pos} - P_t^{neg}| \tag{3.3}$$

The confidence value i_t indicates the level of confidence associated with decision d_t. For example, if $P_t^{pos} = 0.9$ and $P_t^{neg} = 0.1$, then the resulting $i_t = 0.8$, implying a high likelihood that decision d_t is accurate. Conversely, if $P_t^{pos} = P_t^{neg} = 0.5$, i_t becomes 0, indicating a contentious decision d_t. Therefore, we use i_t as the new evidence for the evidence update. If $P_t^{pos} > P_t^{neg}$, it indicates the positive decision is more likely to be correct, so the evidence α_t^x and β_t^x is updated for each advisor using the following equation:

$$\begin{aligned} \forall x \in A_{Y_t}^{pos}, \quad \alpha_{t+1}^x \leftarrow \alpha_t^x + i_t \\ \forall x \in A_{Y_t}^{neg}, \quad \beta_{t+1}^x \leftarrow \beta_t^x + i_t \end{aligned} \tag{3.4}$$

Similarly, if $P_t^{pos} < P_t^{neg}$, the negative option is considered as the correct decision, and so we update α_t^x and β_t^x using:

$$\begin{aligned} \forall x \in A_{Y_t}^{neg}, \quad \alpha_{t+1}^x \leftarrow \alpha_t^x + i_t \\ \forall x \in A_{Y_t}^{pos}, \quad \beta_{t+1}^x \leftarrow \beta_t^x + i_t \end{aligned} \tag{3.5}$$

Then we can update uncertainty θ_{t+1}^x and trustworthiness τ_{t+1}^x by Equation 3.1 and 3.2 based on α_{t+1}^x and β_{t+1}^x.

3.1.3 Binary Bayesian and Weighted Voting Ensemble Decision Model

In this section, we introduce the Binary Bayesian and Weighted Voting Ensemble (BBWVE) decision model, which is the decision function f used within MABSDM. The BBWVE method takes the advice set A_t and the trustworthiness vector $\vec{\tau}_t$ as input, and the output is a probability distribution over two options for decision-making.

In more detail, the BBWVE method combines two common decision-making methods: Bayesian aggregation and weighted voting (see details in Section 2.1). Specifically, when the model lacks sufficient evidence to ascertain the trustworthiness of advisors, the BBWVE method prioritizes the weighted voting method. As the evidence accumulates, the BBWVE method transitions to the Bayesian aggregation method. In what follows, we first explain the Bayesian and weighted voting methods. We then run an experimental evaluation comparing these two methods, showing that the weighted voting method has a higher decision accuracy than the Bayesian aggregation method when the trustworthiness of advisors is unreliable, while the Bayesian aggregation method gradually outperforms the weighted voting method as the reliability of trustworthiness increases, and motivating our combined approach. Finally, we detail the BBWVE method which combines the two approaches.

3.1.3.1 Existing Methods

In this section, we detail two existing methods, the Bayesian aggregation method, and the weighted voting method. They are also parts of the BBWVE method. To begin with, for the Bayesian aggregation method, for each problem t, we recall that A_t denotes the advice set. We recall that d_t^* is the ground truth of problem t. Given the advice set A_t, let $P_t^{b,pos} := P_t^{b,pos}(d_t^* = pos|A_t)$ denote the probability $d_t^* = pos$ obtained through aggregating by the Bayesian aggregation method (b represents the Bayesian aggregation method). Similarly, let $P_t^{b,neg} := P_t^{b,neg}(d_t^* = neg|A_t)$ denote the probability $d_t^* = neg$. We can express $P_t^{b,pos}$ and $P_t^{b,neg}$ as:

$$P_t^{b,pos} = \frac{P_{pos}P(A_t|d_t^* = pos)}{P_{pos}P(A_t|d_t^* = pos) + P_{neg}P(A_t|d_t^* = neg)} \tag{3.6}$$

$$P_t^{b,neg} = \frac{P_{neg}P(A_t|d_t^* = neg)}{P_{neg}P(A_t|d_t^* = neg) + P_{pos}P(A_t|d_t^* = pos)} \tag{3.7}$$

where P_{pos} and P_{neg} are the basic probability. For example, if there is no prior information, the advisor randomly selects the option "positive" or "negative" with the same probability, i.e., P_{pos} and P_{neg} are 0.5. $P(A_t|d_t^* = pos)$ or $P(A_t|d_t^* = neg)$ represents given the condition that the positive or negative option is correct, the probability that the advice set A_t is correct. They can be computed using the trustworthiness τ_t^x mentioned in section 3.1.2. So it can be expressed as:

$$P(A_t|d_t^* = pos) = \prod_{i \in A_{Y_t}^{pos}} \tau_t^i \prod_{j \in A_{Y_t}^{neg}} (1 - \tau_t^j) \tag{3.8}$$

$$P(A_t|d_t^* = neg) = \prod_{i \in A_{Y_t}^{neg}} \tau_t^i \prod_{j \in A_{Y_t}^{pos}} (1 - \tau_t^j) \tag{3.9}$$

In addition, for the weighted voting method, the advisor's trustworthiness τ_t^x is considered weight in the weighted voting method. The correct probability of the positive and negative option can be denoted as $P_t^{w,pos} := P_t^{w,pos}(d_t^* = pos|A_t)$ and $P_t^{w,neg} := P_t^{w,neg}(d_t^* = neg|A_t)$, respectively ($w$ represents the weighed voting method). These two probability can be expressed as:

$$P_t^{w,pos} = \frac{\sum_{i \in A_{Y_t}^{pos}} \tau_t^i}{\sum_{j \in Y_t} \tau_t^j} \tag{3.10}$$

$$P_t^{w,neg} = \frac{\sum_{i \in A_{Y_t}^{neg}} \tau_t^i}{\sum_{j \in Y_t} \tau_t^j} \tag{3.11}$$

3.1.3.2 BBWVE Motivation Experiment

We conduct a comparative experiment between the Bayesian aggregation method and the weighted voting method to explain the motivation behind the design of the BB-WVE method. Specifically, we examine the performances of the Bayesian aggregation method and the weighted voting method under varying levels of trustworthiness reliability across 6 conditions.

To initiate the experiment, we create 50 simulated advisors. Each advisor is assigned a real accuracy value, represented as τ_x^r, which is sampled from an *Extended Rectified Gaussian distribution* (ERGd) (Palmer et al., 2017). The reason for using ERGd is that it is possible to set different means and standard deviations to simulate the different distributions of the real accuracy of advisors to comprehensively evaluate our method. For instance, if $\tau_x^r = 0.8$, it indicates that advisor x has an 80% probability of choosing the correct option, while there is a 20% probability that advisor x randomly selects the incorrect options.

In addition, for each advisor x, we generate a trustworthiness τ_x^u for practical use by both the Bayesian aggregation method and the weighted voting method. Additionally, we randomly generate a probability variable $ran_x \in [0, 1]$. This allows us to calculate τ_x^u as follows: $\tau_x^u = ran_x + (\tau_x^r - ran_x) * (dis/100)$, where dis is the distance coefficient that is used to control the error between τ_x^u and τ_x^r. As dis increases, τ_x^u gradually approaches the real accuracy τ_x^r from a random value ran_x, which simulates the trustworthiness of advisors transitioning from unreliable to reliable. Moreover, at each dis that is from 0 to 99 (increases by 1 each time), the advisors need to complete $10,000$ problems using the Bayesian aggregation method and the weighted voting method for aggregating. In addition, the number of problems that are correctly decided is the result of the experiment.

Furthermore, we use $\bar{y}$ to represent the average number of advisors per problem in each experiment. For each problem, we randomly select $\bar{y} = 5$ advisors on average from all advisors. In addition, μ and σ denote the mean and standard deviation of the ERGd distribution used to generate τ_x^r. In different conditions, we set $\mu = 0.6, 0.7, 0.8$, and $\sigma = 0.2, 0.3$, respectively. In the experiments, we only consider the case where μ is greater than 0.5. The reason is that advisors' trustworthiness can be misled by advisors whose real trustworthiness is less than 0.5 because we have no ground truth and prior information. In addition, we do not choose $\mu > 0.8$ as the setting because the considerably high μ results in near-perfect decision accuracy, obscuring the nuanced differences between the two decision-making methods.

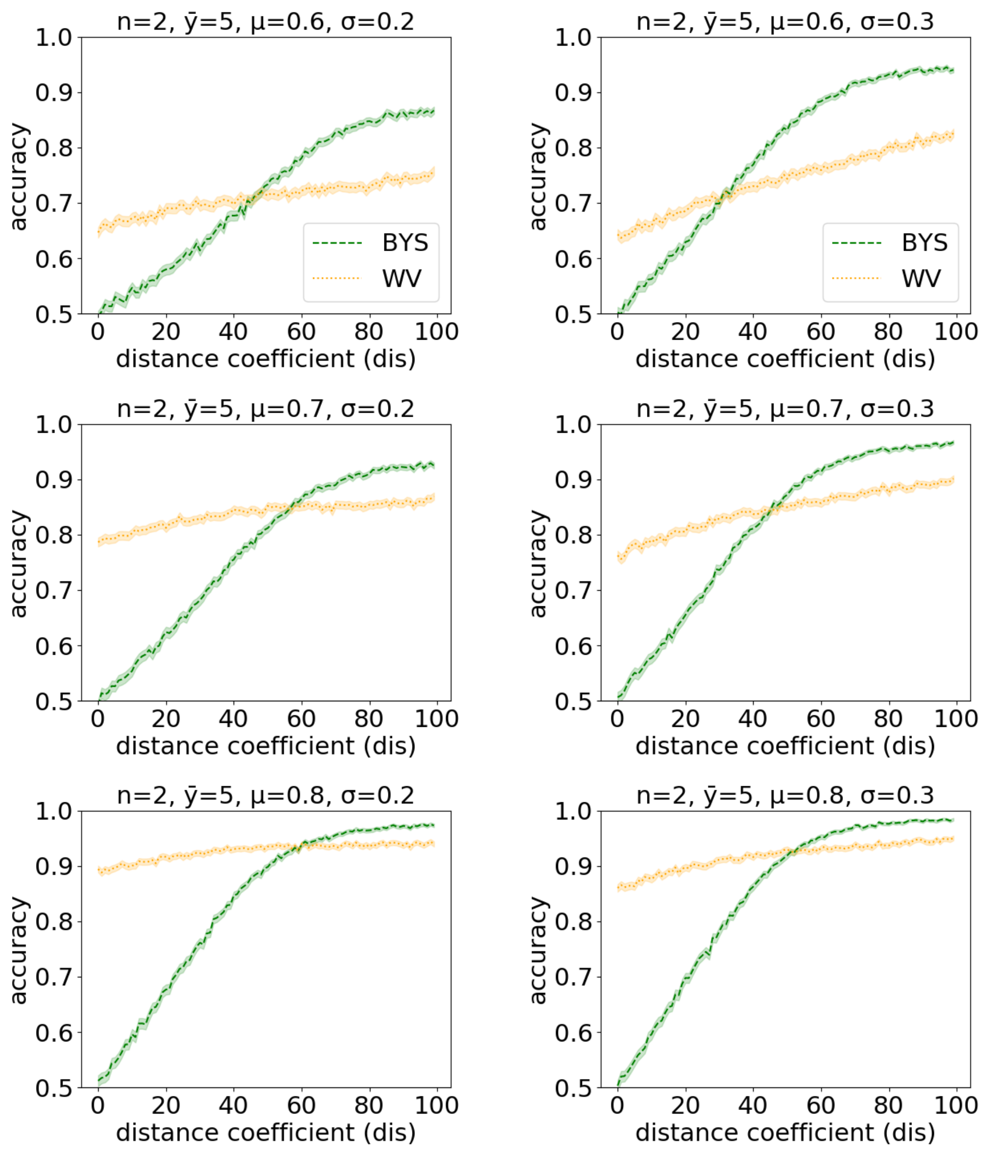


Figure 3.2 These figures show the results of the Bayesian aggregation method and the weighted voting method comparison experiments at different conditions in binary decision-making problems. The X-axis of the figures depicts the distance coefficient *dis*, ranging from 0 to 100, while the Y-axis displays the average accuracy of two methods observed over 10, 000 problems. n is the number of options. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

Figure 3.2 presents six sets of results from the comparison experiments. In these results, the weighted voting method generally outperforms the Bayesian aggregation method when *dis* is small. However, as *dis* increased, the Bayesian aggregation method demonstrate superior performance over the weighted voting method. Therefore, we combine the Bayesian aggregation method and the weighted voting method by the uncertainty of advisors' trustworthiness, to adapt to the varying reliability of advisors' trustworthiness and improve the overall aggregation performance.

3.1.3.3 Ensemble Decision-Making

The BBWVE method combines the Bayesian aggregation method and the weighted voting method by using the average uncertainty of advisors' trustworthiness as a weight to control the trust placed in each method. This uncertainty indicates the reliability of the present trustworthiness of advisors. Let $\bar{\theta}_t \in (0, 1]$ denote the average uncertainty of the advisor set Y_t, and let $|Y_t|$ denote the cardinality of Y_t, i.e., the number of advisors in Y_t. Let θ_t^i denote the uncertainty of advisor i. $\bar{\theta}_t$ can be expressed as:

$$\bar{\theta}_t = \frac{\sum_{i \in Y_t} \theta_t^i}{|Y_t|} \tag{3.12}$$

During the initialization stage, the BBWVE method places more trust in the weighted voting method due to the low reliability of the advisors' trustworthiness. As evidence accumulates and the average uncertainty decreases, the reliability of the trustworthiness increases. This results in a higher weight being assigned to the Bayesian aggregation method, gradually shifting the balance of trust from the weighted voting method to the Bayesian aggregation method.

From ensemble method, given the advice set A_t, the probability that $d_t^* = pos$ and $d_t^* = neg$ can be denote as $P_t^{pos} := P_t^{pos}(d_t^* = pos|A_t)$ and $P_t^{neg} := P_t^{neg}(d_t^* = neg|A_t)$, which can be expressed as:

$$P_t^{pos} = (1 - \bar{\theta}_t)P_t^{b,pos} + \bar{\theta}_t P_t^{w,pos} \tag{3.13}$$

$$P_t^{neg} = (1 - \bar{\theta}_t)P_t^{b,neg} + \bar{\theta}_t P_t^{w,neg} \tag{3.14}$$

As $\bar{\theta}_t$ decreases, the weights of the weighted voting methods decrease, and the weights of the Bayesian aggregation methods increase. For any P_t^{pos} and P_t^{neg}, we have $P_t^{pos} + P_t^{neg} = 1$.

After aggregating the advice, the system needs to compare P_t^{pos} and P_t^{neg}. The option with the bigger probability is the final decision of problem t, i.e., $d_t = \underset{o \in \{pos, neg\}}{\arg\max}\ P_t^o$. If $P_t^{pos} = P_t^{neg}$, the decision is randomly made between two options.

3.1.4 MABSDM

The MABSDM method is described in Algorithm 1. Line 1 describes the initialization of parameters, including $\vec{\tau}$, $\vec{\theta}$, t_{max}, $\vec{\alpha}$, $\vec{\beta}$, t, where $\vec{\tau}$, $\vec{\theta}$, $\vec{\alpha}$, and $\vec{\beta}$ are the vectors contain trustworthiness parameters of all advisors, and t_{max} is the maximum number of problems. First, we obtain the answer set $A_{Y_t}^{pos}$ and $A_{Y_t}^{neg}$ (Line 3). Then we calculate P_t^{pos}, P_t^{neg}, i_t by Equation 3.13, 3.14, 3.3, respectively (Line 4). Next, the decision d_t is obtained according to P_t^{pos} and P_t^{neg}. If $P_t^{pos} > P_t^{neg}$, the decision d_t is pos (Line 5-6). In contrast, the decision d_t is *neg* if $P_t^{pos} < P_t^{neg}$ (Line 7-8). Then, we can use the confidence value i_t to update the evidence vector $\vec{\alpha}$ and $\vec{\beta}$ by Equation 3.4 and 3.5. Subsequently, the trustworthiness vector $\vec{\tau}$ and the uncertainty vector $\vec{\theta}$ can be updated using Equation 3.1 and 3.2 (Line 9). Then, we can make the decision of the next problem $t+1$, until t exceeds the maximum number of problems t_{max} (Line 11).

Algorithm 1 MABSDM algorithm

```
1:  initialize $\vec{\tau}, \vec{\theta}, t_{max}, \vec{\alpha} = \vec{\beta} = 0, t = 0$
2:  while true do
3:      obtain $A_{Y_t}^{pos}, A_{Y_t}^{neg}$
4:      calculate $P_t^{pos}, P_t^{neg}, i_t$
5:      if $P_t^{pos} > P_t^{neg}$ do
6:          $d_t = pos$
7:      else do
8:          $d_t = neg$
9:      update $\vec{\alpha}, \vec{\beta}, \vec{\tau}, \vec{\theta}$
10:     $t = t + 1$
11: until $t > t_{max}$
```

3.2 Experiments

In this section, we design a problem-advice experiment, which is used to evaluate the performance of MABSDM. Specifically, we compare MABSDM with five methods that are combined with different aggregation methods: Bayesian aggregation (BYS) (see Section 3.1.3.1), Weighted Voting (WV) (see Section 3.1.3.1), Binary Bayesian and Weighted Voting Ensemble (BBWVE) (see Section 3.1.3), and trustworthiness modeling methods: Beta distribution trustworthiness model (Beta) (see Section 2.2.1), Binary Cautious Trustworthiness model (BCT) (see Section 3.1.2). See Table 3.1 for the overview. Then, the combined models are (1) BYS-Beta; (2) WV-Beta; (3) BBWVE-Beta; (4) BYS-BCT; (5) WV-BCT, and our method MABSDM consists of BBWVE and BCT.

Table 3.1 the abbreviations of methods in MABSDM experiments

abbreviations list	
BYS	Bayesian Aggregation
WV	Weighted Voting
BBWVE	Binary Bayesian and Weighted Voting Ensemble
Beta	Beta Distribution Trustworthiness Model
BCT	Binary Cautious Trustworthiness Model

In addition, we use statistical hypothesis tests on the experimental results. The Mann–Whitney U test (Mann and Whitney, 1947) combined with Bonferroni Correction (Bonferroni,

1936) is used to test the difference between the results of MABSDM with five baseline methods, respectively.

3.2.1 Setting

In this problem-advice experiment, we generate 50 simulated advisors with a different real accuracy for each experiment. For each problem, the system randomly selects 5 advisors on average from 50 advisors to give advice to problems. For each problem, the system randomly selects 5 advisors on average from 50 advisors to give advice to problems. For this setting, if we select fewer than 5 advisors for each problem, it is difficult to observe the performance of different aggregation methods. In addition, if we select many advisors, e.g., 50 advisors, the decision accuracy of all methods is very high, blurring the distinctions between individual strategies. In addition, recall that τ_x^r denotes the real accuracy of advisor x. In order to observe the impact of the average real accuracy of advisors on performance, for each advisor x, τ_x^r is sampled from ERGd whose mean μ ranges in the set $\{0.51, 0.52, ..., 1\}$. In addition, to simulate the different trustworthiness distributions, we design 6 groups of experiments with different standard deviations $\sigma = \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$. The advisors need to aggregate their advice to make decisions on $1,000$ problems sequentially in each experiment. Table 3.2 presents the setting of the experiments in simulated environments.

In addition, after each feedback, the system can aggregate advice through six different methods. Then, the aggregated advice is compared to the ground truth. If the aggregated advice is the same as the ground truth, then the advice is considered correct. Otherwise, it is considered an error. After the feedback of $1,000$ problems, we can know how many correct decisions there are, and the accuracy of different methods can be obtained. Moreover, to reduce the influence of the randomness, we run each experiment with the same settings $1,000$ times (i.e., using the same mean and standard deviation to generate the advisors' real accuracy, and all methods use the same set of advisors in the same experiment).

Table 3.2 the setting of MABSDM experiments

setting	value
the mean of advisors' real accuracy μ	0.51, 0.52,..., 1
the standard deviation of advisors' real accuracy σ	0, 0.1, 0.2, 0.3, 0.4, 0.5
total number of advisors	50
average number of advisors of each problem $\bar{y}$	5
the number of problems in each problem set	$1,000$
repeat running time of each setting	$1,000$

Table 3.3 This table presents the mean and standard deviation of the accuracy of six methods in simulated experiments. The mean accuracy is calculated over all results ($50 * 1,000$ results). The standard deviation is computed from 1,000 repeated experiments.

σ	**MABSDM**	**BBWVE -Beta**	**BYS -BCT**	**BYS -Beta**	**WV -BCT**	**WV -Beta**
$\sigma = 0$	0.8167 ±0.0017	0.8020 ±0.0025	0.7535 ±0.0143	0.7420 ±0.018	0.8287 ±0.0015	0.8283 ±0.0014
$\sigma = 0.1$	0.8351 ±0.003	0.8212 ±0.0039	0.7701 ±0.0168	0.7571 ±0.0196	0.8359 ±0.0027	0.8357 ±0.8422
$\sigma = 0.2$	0.8725 ±0.0075	0.8611 ±0.0095	0.8041 ±0.0238	0.7863 ±0.0267	0.8408 ±0.0049	0.8422 ±0.0049
$\sigma = 0.3$	0.8955 ±0.0151	0.8863 ±0.0174	0.8262 ±0.0334	0.8052 ±0.0353	0.8388 ±0.0074	0.8428 ±0.0076
$\sigma = 0.4$	0.9008 ±0.0208	0.8911 ±0.0255	0.8270 ±0.0387	0.8069 ±0.0420	0.8332 ±0.0102	0.8394 ±0.0106
$\sigma = 0.5$	0.8972 ±0.0259	0.8880 ±0.0299	0.8192 ±0.0425	0.8032 ±0.0468	0.8257 ±0.0129	0.8339 ±0.0135

3.2.2 Results

Table 3.3 shows the results in each group of experiments. Overall, MABSDM exhibits the best performance in almost all environments. Specifically, MABSDM achieved the best performance in 6 groups of experiments, while only slightly lower than weighted voting methods when $\sigma = 0, 0.1$. In addition, MABSDM outperforms BBWVE-Beta in all sets of experiments, which indicates our BCT can further improve the performance of BBWVE. Moreover, BBWVE methods have better stability than Bayesian methods, because the standard deviations of BBWVE methods are lower than Bayesian methods. In the following, we analyze the results of each group of experiments in detail.

First of all, Figure 3.3 shows the curve of the average accuracy (left) and the trustworthiness error (right) of six methods as the mean of real accuracy μ increases where the standard deviation of advisors' trustworthiness σ is 0, i.e., all the advisors have the same accuracy. In more detail, in terms of accuracy results, the two weighted voting methods have the best performance, while MABSDM is a little worse than them and Bayesian methods perform worst. Moreover, for the ability to model trustworthiness, WV-Beta has the best performance the average error is only 0.1, whereas MABSDM is a little worse than it. In contrast, BYS-Beta, BYS-BCT, and WV-BCT cannot model trustworthiness well, because their error is around 0.13.

However, in practice, assuming that all advisors have the same trustworthiness is unrealistic. In addition, advisors with the same trustworthiness greatly weaken the role of trustworthiness in the aggregation method, and the same performance can be achieved by using the majority voting method without the trustworthiness model.

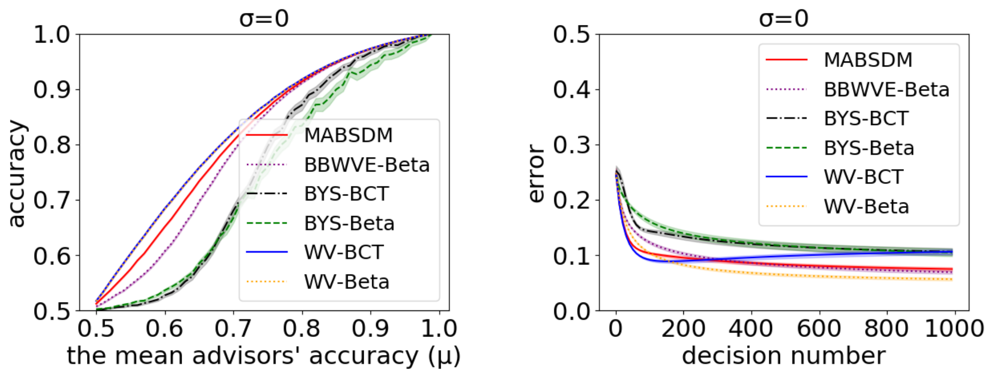


Figure 3.3 These figures show the accuracy curves (left) and trustworthiness error curves (right) where the standard deviation σ between advisors is 0. In the left figure, the X-axis of the figures depicts the mean of real accuracy of advisors μ, ranging from 0.51 to 1, while the Y-axis displays the average accuracy of six methods observed over 1,000 experiments. In the right figure, the X-axis of the figures describes the decision number, while the Y-axis displays the average error of six methods observed in all ranges of real advisor's accuracy over 1,000 experiments (average value of $50 * 1000$ results). The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

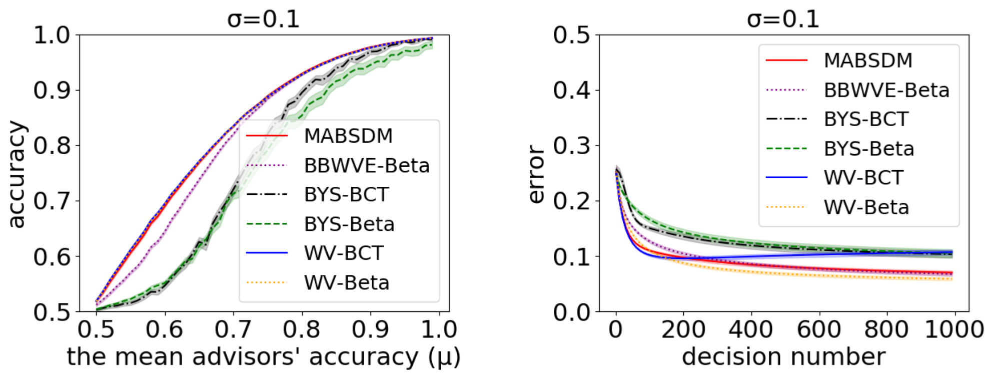


Figure 3.4 These figures show the accuracy curves (left) and trustworthiness error curves (right) where the standard deviation σ between advisors is 0.1. More details see in Figure 3.3.

Furthermore, Figure 3.4 illustrates the curves with σ set to 0.1. Specifically, MABSDM, WV-BCT, and WV-Beta exhibit similar decision accuracy, while BBWVE-Beta slightly lags behind them. Additionally, WV-Beta outperforms the other methods in terms of model trustworthiness, with MABSDM as the second-best, showing only a marginal difference from WV-Beta. Moreover, similar to the results obtained when $\sigma = 0$, Bayesian methods demonstrate the poorest performance in both decision accuracy and trustworthiness modeling. However, the differences in the trustworthiness of advisors are not big when $\sigma = 0.1$, so BBWVE does not exhibit an advantage in this scenario.

Moreover, Figure 3.5 depicts the curves for σ values of 0.2 and 0.3. MABSDM has the best performance in decision accuracy and trustworthiness modeling, while the

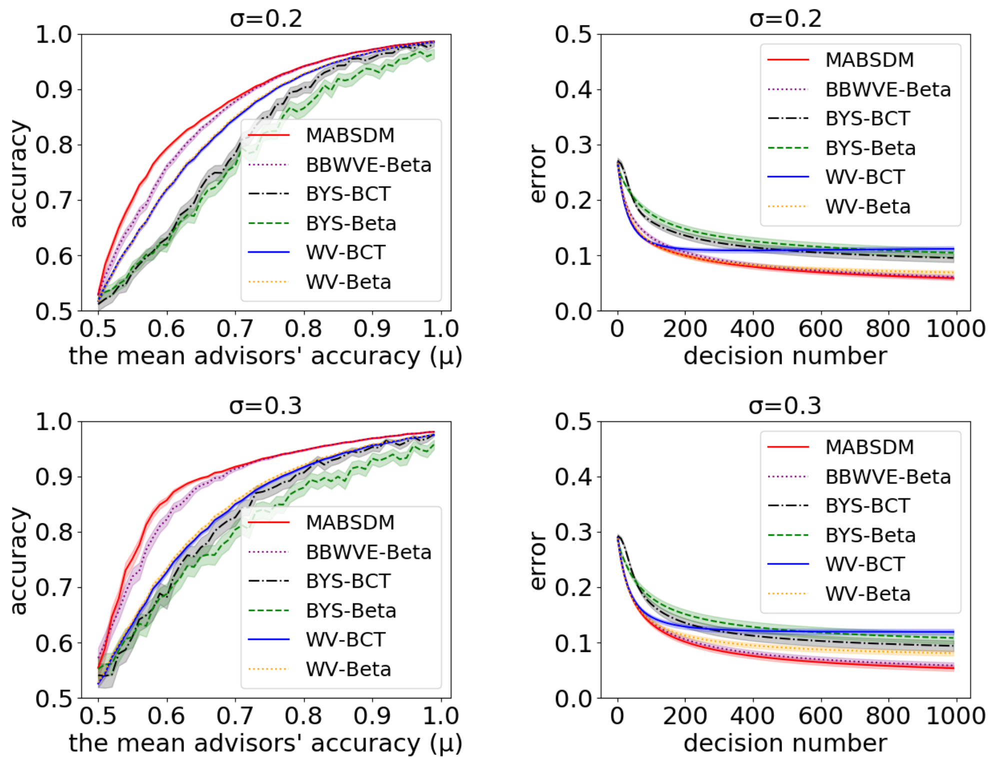


Figure 3.5 These figures show the accuracy curves (left) and trustworthiness error curves (right) where the standard deviation σ between advisors are 0.2 and 0.3. More details see in Figure 3.3.

BBWVE-Beta is a little worse than MABSDM. Notably, MABSDM demonstrates a significant advantage when the real accuracy of advisors is low. Particularly, at $\sigma = 0.3$, the decision accuracy of MABSDM is approximately 10% higher than that of Bayesian methods and weighted voting methods, where μ ranges from 0.55 to 0.75. Additionally, as μ increases above 0.65, the advantage of MABSDM gradually diminishes. This is expected since all methods tend to perform well when advisors have high accuracy. As μ increases, the differences between different decision methods decrease, as the upper bound of accuracy is 1. In conclusion, the results indicate that MABSDM excels in scenarios with low advisor accuracy, but its advantage diminishes as advisor accuracy improves.

In addition, Figure 3.6 shows the curves where σ are 0.4 and 0.5. Specifically, MABSDM still has the best performance in decision accuracy and trustworthiness modeling, while the BBWVE-Beta follows closely. In addition, in general, weighted voting methods outperform Bayesian methods, but the gap between them decreases as μ increases.

Finally, we conducted paired hypotheses tests using the Mann–Whitney U test combined with Holm Bonferroni Correction to compare the accuracy results of MABSDM and the five other methods. Specifically, the p-values revealed that out of a total of $1,500$ comparisons ($50 \times 6 \times 5 = 1,500$). Among them, $1,084$ pair results show significant differences, which supports our results. In addition, we observe that BBWVE-Beta

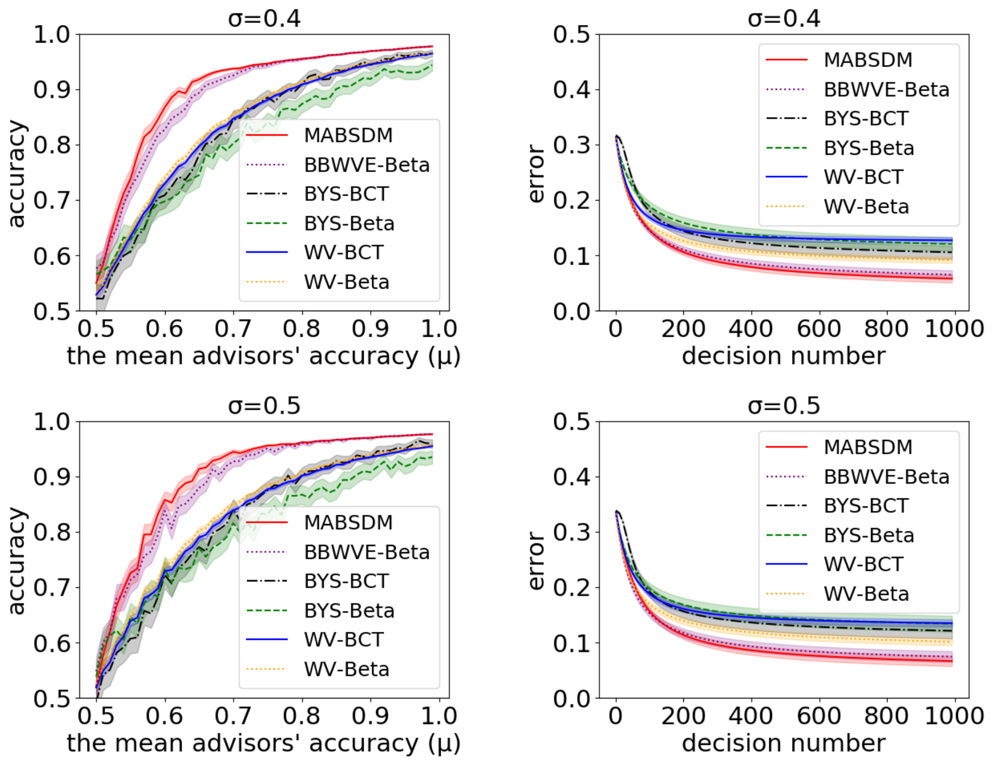


Figure 3.6 These figures show the accuracy curves (left) and trustworthiness error curves (right) where the standard deviation σ between advisors are 0.4 and 0.5. More details see in Figure 3.3.

and MABSDM have only a few results that are significantly different, where $\sigma = 0.3$ and 0.4. Moreover, most of the results of MABSDM and Bayesian methods have significant differences. Whereas the results between MABSDM and Weighted Voting methods have significant differences where $\sigma > 0.1$. This is consistent with the results obtained in our experiments.

3.3 Summary

In this chapter, we complete Objective **O1.1**, **O1.2** (see Section 1.3.1). We propose MABSDM, which is a multi-advisor binary sequential decision-making method. In more detail, MABSDM considers (1) modeling the advisors' trustworthiness sequentially without prior information and ground truth and, (2) making optimal decisions by the advice of multiple advisors. Moreover, the results show that our MABSDM method outperforms the benchmarks using state-of-the-art methods including weighted voting, the Bayesian aggregation method, and the Beta distribution trustworthiness model.

Chapter 4

Multi-Advisor Interactive Reinforcement Learning

The previous chapter proposes the MABSDM method for binary decision-making problems. In this chapter, we extend the application of MABSDM to a multi-advisor IRL system. Specifically, the multi-advisor IRL shares five key similarities with the multi-advisor sequential decision-making problem. First, MAIRL involves aggregating advice of multiple advisors to provide human rewards (make decisions on the reward provided to the agent). Second, due to reward sparsity, advisors often lack access to the ground truth on whether rewards are reliable. Third, prior information on advisors' trustworthiness is often unavailable, necessitating the development of trustworthiness models over time. Fourth, IRL is a sequential decision-making problem because the advisors need to provide rewards to the agent at each time point. Lastly, we use binary feedback as the feedback form in our IRL system (see Section 4.1.2), aligning with the binary decision-making focus of MABSDM.

In this chapter, in addition to using MABSDM to give rewards to the agent, MAIRL considers reducing the human feedback burden and improving the utilization efficiency of advice. To address the Objective **O2.1** and **O2.2**, in this chapter, we propose a novel interactive reinforcement learning system called Multi-Advisor Interactive Reinforcement Learning (MAIRL). Specifically, MAIRL can aggregate the binary advice of multiple imperfect advisors into a reliable reward for agent training in a reward-sparse environment. In addition, the review model in MAIRL can correct the unreliable reward from historical feedback. Moreover, we use the binary feedback form in MAIRL. In particular, our binary feedback experiments show that binary feedback outperforms other feedback forms including ranking feedback, scaling feedback, and state value feedback. Finally, we conduct grid-world experiments to show that the policy trained by the MAIRL with the review model is closer to the optimal policy than that without a review model.

The rest of the chapter is structured as follows. First, we describe the MAIRL method in Section 4.1. Second, we present the experiments and results in Section 4.2. Lastly, we summarize our work in Section 4.3.

4.1 Model Description

MAIRL can aggregate the binary feedback of multiple advisors into a reward to help the agent train, which greatly improves the learning efficiency of the agent. Within the framework of MAIRL, we utilize the BCT model as the trustworthiness model (see Section 3.1.2) and employ the BBWVE method to build the decision model (see Section 3.1.3).

In addition to employing MABSDM for deciding the rewards, MAIRL considers reducing the feedback burden of advisors. First, to reduce the cognitive burden, MAIRL uses binary feedback as the feedback form. Under this setup, the advisors are only required to provide ”positive” or ”negative” assessments of the performance of the agent. In addition, our experimental results show that binary feedback outperforms other feedback forms. Second, to increase the utilization efficiency of human advice, MAIRL maintains a record of historical advice, which is leveraged to assign rewards to state-action pairs that were previously subjected to feedback. In particular, the review model in MAIRL can assess the reliability of historical rewards and decide whether to query more advisors to correct the unreliable ones. Figure 4.1 shows its structure.

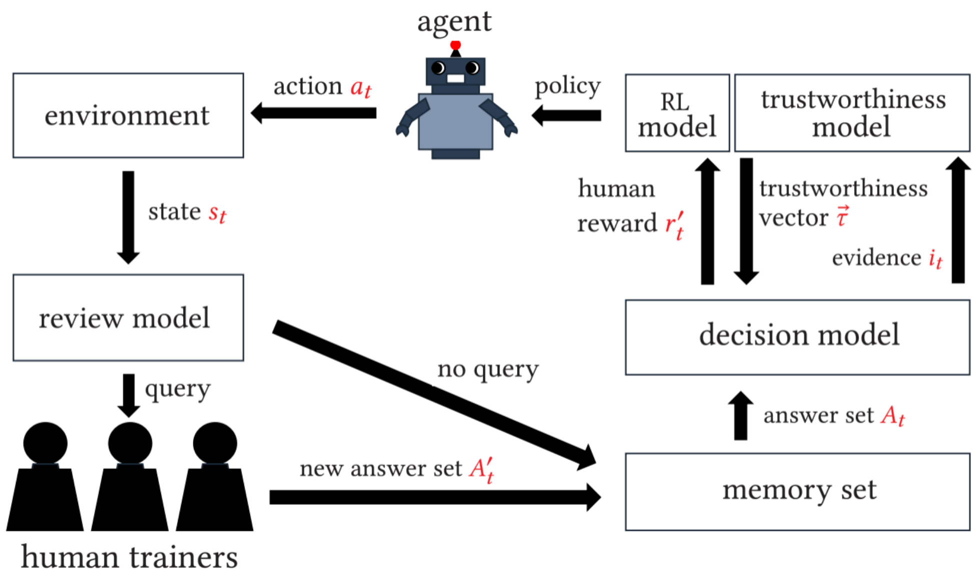


Figure 4.1 The review model receives the state from the environment and decides whether to query advisors according to historical feedback. Then, the decision model decides the final reward given to the agent by the advisors’ advice and trustworthiness. Lastly, the RL model updates the policy with human rewards and the trustworthiness model updates the trustworthiness of advisors.

4.1.1 Problem Formalization

In this chapter, we present a multi-advisor IRL system that aggregates the advice of multiple imperfect advisors to decide rewards for the agent's training. More formally, the RL model is represented as a Markov Decision Process (MDP) with five key elements: $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma \rangle$, where $\mathcal{S}$ represents the set of states (state space), $\mathcal{A}$ denotes the action space, $\mathcal{T}$ is the state transition function, and R represents the reward function. Furthermore, the discount coefficient $\gamma \in [0,1]$ indicates the influence of future rewards on the current state value. At each time t, the agent observes a state $s_t \in \mathcal{S}$. Then, it selects an action $a_t \in \mathcal{A}$ by policy $\pi(a_t|s_t)$; After this, the agent receives a reward r_t from the reward function R such that $r_t = R(s_t, a_t)$. In IRL, we denote the rewards provided by human advisors as r'_t. In this Section, we use human reward r'_t instead of the environmental reward r_t.

In multi-advisor IRL, at each time t, a set of advisors $Y_t \subseteq X$ gives advice according to the state action pair (s_t, a_t). If an advisor believes the action taken by the agent in the present state is the best action, then a *pos* advice is given. In contrast, an advisor gives a *neg* advice if he or she believes the action is not the best action. In addition, for tasks with continuous action spaces, the actions need to be discretized before applying MAIRL. At each time t, we obtain the advice set $A_t = \{A^{pos}_{Y_t}, A^{neg}_{Y_t}\}$.

In addition, advisors' rewards can be incorrect. Therefore, the model needs to consider the trustworthiness of advisors. Similar to Section 3.1.1, we denote with $\vec{\tau}_t$ the vector containing all the advisors' trustworthiness values. Therefore, the system needs to use a decision function to aggregate the answers to give the human reward r'_t. Specifically, we use $r'_t = f(A_t, \vec{\tau}_t) \in \{pos, neg\}$ to refer to the decision-making function. Let r^*_t denote the correct reward, which is the best reward for agent training among the rewards of all actions in state s_t. If $r'_t = r^*_t$, we consider that the reward is correct. Let ν denote the number of correct rewards, i.e., $\nu = |\{t \in T | r^*_t = f(A_t, \vec{\tau}_t)\}|$. Our goal is to maximize the number of the correct rewards ν.

4.1.2 Binary Feedback

MAIRL uses binary feedback $r'_t \in \{pos, neg\}$ as the feedback form. At the current state, if the human believes the action taken by the agent is the best, then a positive reward *pos* is given. Otherwise, a negative reward *neg* is given. There are three reasons for choosing binary feedback.

First, in the design of feedback forms, one important aim is to minimize the cognitive burden on human advisors. Binary feedback achieves this by presenting advisors with a simpler choice between two options, rather than requiring them to select from multiple options or scores. This approach is intended to make the feedback process more user-friendly for advisors, facilitating their participation in the training of the RL agent.

Second, binary feedback is beneficial for reducing feedback delay. In the context of IRL, feedback delay refers to the time taken by a human advisor to provide a reward after receiving the state-action information, and it is recognized as a challenging issue (Knox and Stone, 2008b). Prolonged feedback delay can lead to the human advisor missing the feedback for the correct state-action pair, potentially assigning the reward to an incorrect one. However, utilizing binary feedback simplifies and accelerates the decision-making process for advisors compared to other feedback forms, leading to reduced feedback delay. As a consequence, humans are more likely to provide accurate feedback between two actions when presented with binary options. Therefore, binary feedback both reduces the feedback pressure on humans and ensures the quality of rewards.

Third, our experiments demonstrate that binary feedback outperforms alternative feedback forms (see Section 4.2.2). To be specific, agents trained by binary feedback are closer to the optimal strategy than those trained by other feedback forms. In RL, the agent's primary task is to discern the optimal action within each state compared to the other available actions, thus rendering additional information inconsequential for accelerating the learning of the optimal strategy. Moreover, the results show that binary feedback exhibits greater robustness to incorrect feedback.

4.1.3 Review Model

In IRL, the agent is likely to encounter the same state-action pair at different times. If we only rely on MABSDM, advisors are required to give advice endlessly. In IRL, advisor feedback is often expensive and time-consuming, so unlimited query human advisors are unrealistic. Therefore, we need to reduce the pressure of querying humans as much as possible. To solve this problem, MAIRL uses a memory system, which can record the advice of historical feedback. If the agent re-encounters the same state-action pair, MAIRL uses the reward in the memory, greatly reducing the number of human feedback.

However, the problem with doing this is that the future rewards are wrong if the previous reward is wrong. Therefore, to guarantee reward accuracy, a review mechanism is applied in the MAIRL system to correct the wrong rewards. The system re-asks those state-action pairs that have unreliable rewards. The MAIRL system is described in Algorithm 2. Line 1 describes the initialization of parameters, including state s, action a, the maximum number of training episodes n_{max}, and maximum steps T.

Algorithm 2 MAIRL system algorithm

```
1: initialize $\forall s \in \mathcal{S}, a \in \mathcal{A}(s), n_{max}, T$
2: while true do
3:     initialize state $s_0$
4:     $a_0 \leftarrow argmax_a(RLmodel(s_0))$
5:     for $step = 1$ to $T$ do
6:         take action $a_t$, obtain $s_{t+1}$
7:         calculate $P_t^{pos}(r_t^* = pos|A_{s,a}^{pos}, A_{s,a}^{neg}), P_t^{neg}(r_t^* = neg|A_{s,a}^{pos}, A_{s,a}^{neg})$, $i'_{s,a}$, $r'_{s,a}$,
   $P_{re}(s_t, a_t)$, generate random number $P_t^q \in [0,1]$
8:         if $P_t^q < P_{re}(s_t, a_t)$ do
9:             query advisors and obtain $A_{Y_t}^{pos}$, $A_{Y_t}^{neg}$
10:            $A_{Y_t}^{pos} \leftarrow A_{s,a}^{pos} \cup A_{Y_t}^{pos}$
11:            $A_{Y_t}^{neg} \leftarrow A_{s,a}^{neg} \cup A_{Y_t}^{neg}$
12:            recalculate $P_t^{pos}, P_t^{neg}, i_t$
13:            $A_{s,a}^{pos} \leftarrow A_{Y_t}^{pos}$
14:            $A_{s,a}^{neg} \leftarrow A_{Y_t}^{neg}$
15:            if $P_t^{pos} > P_t^{neg}$ do
16:                $r'_t \leftarrow r_{pos}$
17:            else do
18:                $r'_t \leftarrow r_{neg}$
19:        else do
20:            $i_t \leftarrow i'_{s,a}$
21:            $r'_t \leftarrow r'_{s,a}$
22:        $RLmodel.update(s_t, a_t, r'_t)$
23:        $s_t \leftarrow s_{t+1}; a_t \leftarrow a_{t+1}$
24:        update $\vec{\tau}$
25:        $i_{s,a} \leftarrow i_t$
26:    until task end
27: until $n > n_{max}$
```

Let $A_{s,a}^{pos}$ and $A_{s,a}^{neg}$ represent the historical sets of positive and negative advice for the state-action pair (s, a), respectively. At the time t, if the agent encounters a state-action pair that has been previously queried, it needs to decide whether to ask the advisors again (Line 8). Let $P_t^{pos}(r_t^* = pos|A_{s,a}^{pos}, A_{s,a}^{neg})$ and $P_t^{neg}(r_t^* = neg|A_{s,a}^{pos}, A_{s,a}^{neg})$ denote the probability of option *pos* or *neg* is correct that based on historical advice. In this context, due to the trustworthiness changing over time, the system needs to calculate the probability $P_t^{pos}(r_t^* = pos|A_{s,a}^{pos}, A_{s,a}^{neg})$ and $P_t^{neg}(r_t^* = neg|A_{s,a}^{pos}, A_{s,a}^{neg})$ using the present trustworthiness value.

Consequently, the system obtains the confidence value $i'_{s,a}$ and the reward $r'_{s,a}$. The probability of review $P_{re}(s_t, a_t) \in [0, 1]$ can be defined as:

$$P_{re}(s_t, a_t) = 1 - i'_{s,a} \tag{4.1}$$

If $i'_{s,a}$ is larger, it indicates that the historical reward is more reliable, leading to a lower probability of review. Conversely, if $i'_{s,a}$ is smaller, it suggests that the historical reward is less reliable, resulting in a higher probability of review. When the decision is made not to review, the system can utilize the new reward $r'_{s,a}$ as the updated reward r'_t (Line 21).

On the other hand, if the system decides to query again, it needs to take two steps. First, it combines the historical advice with the new advice to create a new advice set $A^{pos}_{Y_t} \leftarrow A^{pos}_{s,a} \cup A^{pos}_{Y_t}$ (Line 10) and $A^{neg}_{Y_t} \leftarrow A^{neg}_{s,a} \cup A^{neg}_{Y_t}$ (Line 11). Second, it recalculates P^{pos}_t and P^{neg}_t to make the new decision. To avoid unnecessary updates, the last update $i_{s,a}$ needs to be removed, and $\vec{\tau}$ is updated with the new evidence i_t (Line 24). Finally, we record the new evidence $i_{s,a} = i_t$ (Line 25).

4.2 Experiments

In this work, we evaluate MAIRL through two sets of experiments. To be specific, we compare binary feedback with three baselines in Section 4.2.2. Second, to evaluate MAIRL, we set up two other multi-advisor IRL methods to compare with our MAIRL system in Section 4.2.3. In the next section, we introduce the experimental design (Section 4.2.1).

4.2.1 Experimental Design

In this section, we present the experimental design, including grid-world environments, the training method, the parameter setting, and the approach to simulating humans.

First, we employ grid-world as the experimental environment in two sets of experiments. It is a classic environment for testing the performance of RL algorithms and has been used in many studies (Coggan, 2004; Tizhoosh, 2005; Arakawa et al., 2018). In addition, to guarantee the generalization ability of the results, we use two kinds of grid-world as the environments. They are cliff grid-world and wall grid-world.

The first type of grid-world, cliff grid-world, comprises five parts: an agent, a start point, an endpoint, normal cells, and cliff cells. As shown in Figure 4.2, the yellow point is the agent, the green cell is the starting point, the blue cell is the endpoint, and the red cell is the cliff. Specifically, in each time step, the agent can select one action from four options (up, down, left, and right). The starting point is the cell where the agent is at the beginning of each game episode. If the agent reaches the endpoint, it passes the game successfully. The goal of the agent is to find the shortest way (taking the minimum number of actions) from the start point to the endpoint. In addition, normal cells can be traversed freely, and the start point is also a normal cell. If the agent reaches the cliff grid, the agent dies, which indicates the task has failed. We construct two sizes

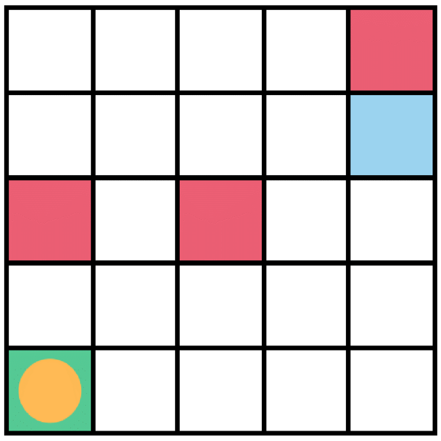


Figure 4.2 An example of $5 * 5$ cliff grid-world. There are 25 cells in total. The green cell is the starting point; the blue cell is the endpoint, and the red cell represents the cliff.

of cliff grid-world, $5 * 5$ and $10 * 10$. These contain 5 and 10 fixed cliff cells, respectively. In each episode, the agents start from a random position, but the endpoint is fixed.

The reason we chose the cliff grid-world is its ability to evaluate the safety performance of the IRL method. Specifically, ensuring the agent's safety is an important aspect of the IRL method, particularly in applications like self-driving cars, where the risk of accidents poses a threat to human life and incurs financial losses. Similarly, in the cliff grid-world, if the agent goes to the cliff cells, it dies.

The second type of environment is the wall grid-world. It consists of 6 parts: an agent, a start point, an endpoint, normal cells, flame cells, and wall cells. Specifically, as shown in Figure 4.3, the yellow point is the agent, the green cell is the starting point, the blue cell is the endpoint, the grey cell is the wall, and the red cell is the flame. Moreover, the same as cliff grid-world, the starting point is where the agent is at the beginning of each game episode. If the agent reaches the endpoint, it passes the game successfully. If the agent reaches the flame grid, it obtains a big punishment, but it cannot die. The wall is an inaccessible place and cannot be passed through. The agent can only go around and find the target. In addition, we also construct two sizes of wall grid-world, $5 * 5$ and 10*10. In $5 * 5$ wall grid-world, we add 2 fixed walls and 3 fixed flame cells, while we add 5 fixed walls and 5 fixed flame cells in 10*10 wall grid-world. In each episode, the agents start from a random position, but the endpoint is fixed.

The reason we use the wall grid-world is that it can evaluate IRL's ability to reduce losses. For the IRL method, reducing training costs is one of its primary goals. For example, in financial market transactions, every failed decision loses a lot of money. It is not lethal, but it can cause losses to the agent.

Second, we employ the *State-Action-Reward-State-Action* (SARSA) algorithm as the basic method to construct IRL methods for our experiments (Rummery and Niranjan, 1994),

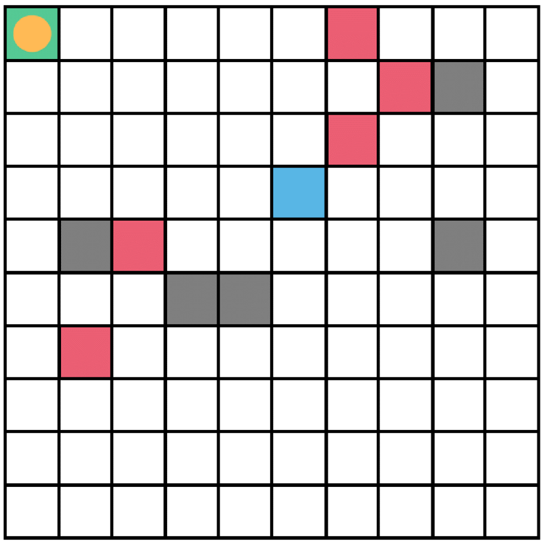


Figure 4.3 An example of $10 * 10$ wall grid-world. There are 100 cells in total. The gray state represents the wall, the green cell is the starting point; the blue cell is the endpoint, and the red cell represents the flame.

(Sutton, 1996). SARSA is an on-policy algorithm known for its stable performance in online control tasks (Thorpe, 1997). In addition, SARSA learns online, meaning it updates its knowledge while interacting with the environment, without the need for a fixed dataset. This fits naturally with the online interaction method of IRL. For our interactive methods, we replace the environmental rewards in SARSA with human rewards provided by advisors.

Let ep denote the number of episodes. The exploration probability is defined as $1/ep$, meaning that as the number of episodes increases, the agent's exploration probability decreases. Initially, during training, the agent's actions are explored randomly. As the agent learns a specific policy, it gradually reduces exploration and focuses on exploiting the learned policy. In addition, regarding the learning rate l and future reward discount coefficient γ, we set them to 0.1 and 0.9, respectively. Through experimentation, we found that these values ensure the SARSA algorithm can converge to the best solution effectively. In different grid-world scenarios, each method uses the same learning rate and discount rate.

Third, we utilize simulated humans as advisors to provide rewards for agents. Specifically, a convergent Q-table trained by the SARSA algorithm is used to simulate humans in the experiments. At each time step t, once the agent takes an action, the simulated human evaluates the action based on its trained Q-table. It then provides a reward to the agent in accordance with the policy in the Q-table. This reward serves as feedback, guiding the agent toward optimal behaviors.

Simulated humans are well-suited for our experimental requirements as we can control their feedback. First, the convergent Q-table obtained from the SARSA algorithm serves as a reliable source to simulate optimal policies for different states in various environments. This guarantees the accuracy of the feedback provided. While real humans might make errors in their judgments due to fatigue or cognitive biases, simulating humans ensures completely accurate feedback. Second, we can add a fixed proportion of incorrect feedback to evaluate the robustness to noise of our method. This assessment is challenging to conduct with real humans whose accuracy is difficult to control.

Fourth, *noise* refers to incorrect feedback from human advisors. It is a critical factor that can limit the performance of IRL. In many real-world tasks, human feedback often is imperfect and contains errors. To reproduce this, our experiments consider the noise scenario. For example, 5% noise indicates that there is a 5% probability that humans provide incorrect feedback.

Specifically, noise indicates that the human gives a reward r'_t that does not match the action a_t, i.e., the noise feedback randomly selects one of the non-original feedback. Take grid-world as an example, there are four actions $\{up, down, left, right\}$ for each state. In state s, the agent takes action up, and up is the only worst action. Therefore, the noise reward of state-action pair (s, up) is randomly selected from the reward of $(s, down)$, $(s, left)$, or $(s, right)$.

4.2.2 Binary Feedback Evaluation

In this section, we evaluate binary feedback by comparing it with three benchmarks in four grid-world environments. In addition, to evaluate the noise robustness of binary feedback, we conduct experiments with 0% and 5% noise, respectively.

Specifically, we conduct experiments using four different feedback forms, binary feedback, scaling feedback, ranking feedback, and state value feedback. Among all forms of feedback, if the agent taking the action can make it closer to the endpoint without dying, then its action is considered optimal. To begin with, we explain four feedback forms used in binary feedback evaluation experiments. First, binary feedback involves humans providing advice in the form of "positive" or "negative" (see Section 2.3.4). Specifically, if the agent selects the best action of the state, it receives a "positive" reward. Otherwise, it receives a "negative" reward. If there are two actions with the same degree of good or bad for the same state, they obtain the same reward. For example, in Figure 4.4, the up and right actions are equally the best actions for a state, so they both receive the highest reward of 0, while the remaining actions receive a reward of -1.

Second, the scaling feedback method is also based on the value of the state-action pair, but the reward setting for each state is not uniform, instead consisting of a range of scores (see Section 2.3.4). The scaling feedback method requires a higher cognitive burden compared to binary feedback, as it involves evaluating the score of each action.

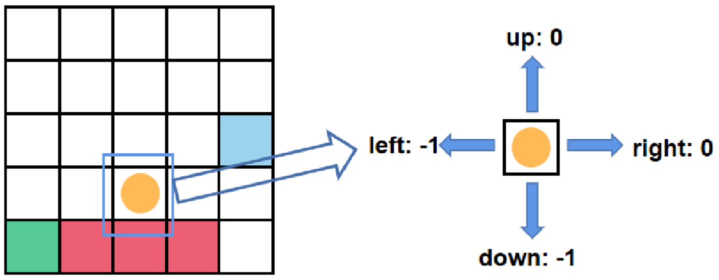


Figure 4.4 The orange dot is the agent, which can obtain a reward of 0 if it moves to the right and up; it can obtain a reward of -1 if it moves down and left.

Figure 4.5 illustrates two examples of how the scaling feedback method works. Similarly to binary feedback, if two actions have an equal positive or negative impact, they will receive the same reward.

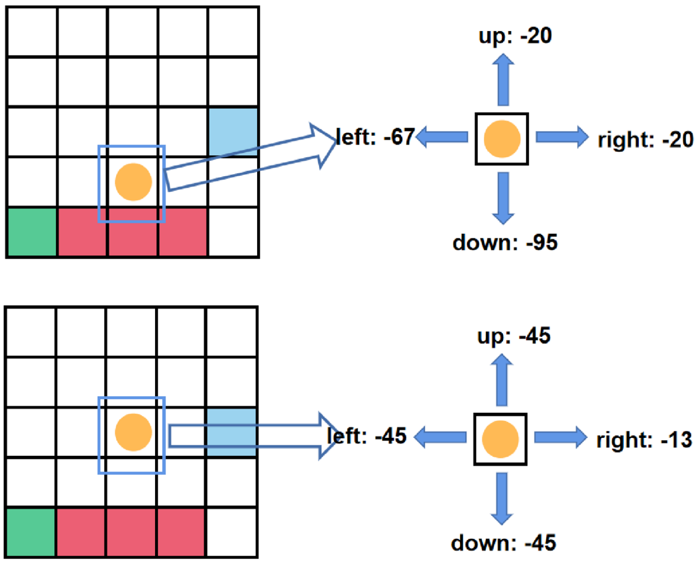


Figure 4.5 Human simulation for scaling feedback method. The orange dot is the agent. In the first figure, the reward for moving right and upward is -20; the reward for moving downward is -95 (because of death); the reward for moving left is -67; in the second figure, The reward for moving to the right is -13; the other actions is the sub-optimal, so the reward is -45.

Third, ranking feedback, inspired by the action advice method (see Section 2.3.3), involves assigning a graded reward based on the value of the state-action pair. Specifically, we first sort all actions for a given state and then provide rewards from high to low based on the action's rank. The cognitive burden of ranking feedback is lower than that of scaling feedback since it only requires ranking actions rather than providing accurate scores for each action. Additionally, compared to binary feedback, the ranking feedback method offers a more nuanced reward for the agent. While binary feedback may not capture sub-optimal rewards, ranking feedback can effectively address this limitation and provide a richer reward signal for the agent. As shown in Figure 4.6,

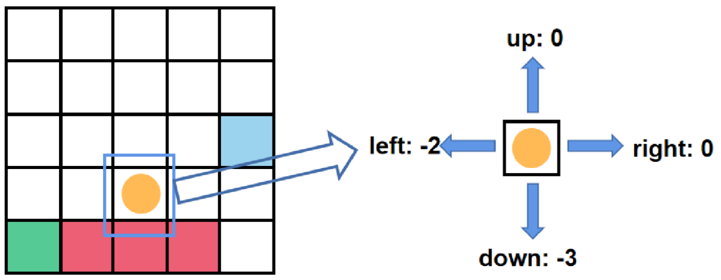
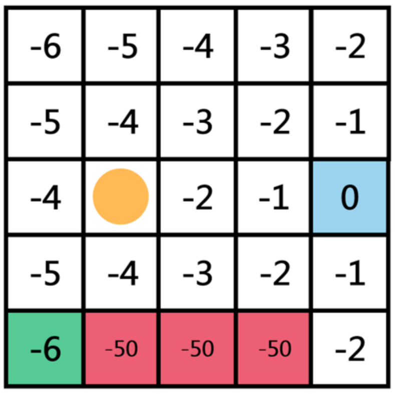


Figure 4.6 The orange dot is the agent, the reward for moving right and upward is 0; the reward for moving downward is -3; the reward for moving left is -2.

in the 5*5 cliff world, the reward can be divided into three levels. The first level is the optimal action that can approach the endpoint. The second level is some sub-optimal action far away from the endpoint. The third level is to fall off a cliff and die after performing the action.

Finally, state value feedback is based on the value of the state where the agent is after taking an action. For example, in Figure 4.7, in a grid-world environment, states closer to the end state have higher values.



-6	-5	-4	-3	-2
-5	-4	-3	-2	-1
-4	●	-2	-1	0
-5	-4	-3	-2	-1
-6	-50	-50	-50	-2

Figure 4.7 The orange dot is the agent, the reward is 0 if the agent arrive the endpoint.

In addition, we set the reward value for different methods. Binary feedback, ranking feedback, and scaling feedback settings are shown in Table 4.1. The first is binary feedback. If the reward of the best action is set to be greater than 0, there is a risk that the agent falls into a loop to obtain the maximum reward (Ng et al., 1999). Additionally, during testing, it was observed that setting the maximum reward to less than 0 is not consistently effective. This causes the agent to spend more time exploring other states if the initial q-value is 0. Therefore, we set the value of "positive" feedback to 0. In terms of other actions, the experiments randomly select the value from the range 0 to -5, 0 to -10, and 0 to -100. Notably, the average results of 100 experiments in different environments do not significantly differ. So we set the reward of "negative" feedback as -1.

Table 4.1 The table shows the assignment of each method to different ranked actions. These values are based on experiments, and the performance of other values is not better than the values in the table.

action ranking	Binary	Ranking	Scaling
best action	0	0	0 to -25
second action	-1	-1	-25 to -50
third action	-1	-2	-50 to -75
worst action	-1	-3	-75 to -100

The second is the ranking feedback method. The setting is shown in Table 4.1. For the best action, the experiments have the same result as binary feedback, and a value of 0 has the best performance. For other actions, we select sequential values for the next three actions in different ranges from 0 to -5, 0 to -10, and 0 to -100. The experimental results also show no significant difference between them. The third is the scaling feedback method. The rewards of the scaling feedback method are set according to the mapping of the converged Q-table. Finally, the state value feedback method uses the value in the convergent state value as a reward. The noise of the state value method is the value randomly selected from three states adjacent to s_t but not the value of the state s_{t+1}.

Table 4.2 episodes number setting

Environment	**Cliff grid-world**		**Wall grid-world**	
Size	**5*5**	**10*10**	**5*5**	**10*10**
Episodes	1,000	5,000	1,000	10,000

Moreover, we set different max episodes for different grid-world environments. Specifically, we use the number of episodes required for SARSA (without human reward) training to converge as the standard for setting max episodes. For example, if the SARSA algorithm takes about $1,000$ episodes to find the best solution in an environment, the max episodes are set to $1,000$. If an IRL method cannot complete training before max episodes that find the best solution, we consider it a failed method. As shown in Table 4.2, 5*5 cliff grid-world is $1,000$ episodes; 10*10 cliff grid-world is $5,000$ episodes; 5*5 wall grid-world is 1,000 episodes; 10* 10 wall grid-world is $10,000$ episodes.

The experimental results are measured using the number of training steps required to find the best solution. Each time the agent moves (takes actions), the number of training steps increases by 1. For example, if binary feedback uses 500 steps to train the agent to find the best solution, the number 500 is the result of binary feedback. Lower numbers of steps indicate better performance. Using the number of training steps has two advantages. First, it accurately reflects the training speed of the agent. Second, it reflects the burden of human feedback, as each step requires simulating human feedback. Hence, the number of steps is equal to the number of human feedback instances.

Table 4.3 The table shows the results in four different grid-world with 0% noise of human feedback. The results are the means and standard deviation of steps used to find the best solution in 100 experiments. "None" means cannot find the best solution. The red results mean that it has the best result among all methods in an environment.

methods	5*5 cliff	10*10 cliff	5*5 wall	10*10 wall
SARSA	2153.35	43641.6	2475.35	31497.15
	+/- 328.95	+/- 4420.46	+/- 460.33	+/- 3659.67
Binary	176.85	2831.25	253.25	2239.65
	+/- 88.07	+/- 1506.23	+/- 104.5	+/- 802.63
Ranking	213.55	2781.65	281.25	2067.25
	+/- 100.65	+/- 1620.67	+/- 88.03	+/- 687.21
Scaling	1455	None	1212.25	24605.1
	+/- 541.32		+/- 504.89	+/- 5366.55
State Value	542.9	19972.8	555.5	17173.65
	+/- 193.08	+/- 2659.66	+/- 459.70	+/- 6717.52

In addition, to determine whether the agent has found the best solution, the system compares the agent's policy with the optimal policy for all states at each time step. If the agent's action matches the optimal policy in each state or matches one of the best actions in states with multiple optimal actions, it is considered to have found the best action. When the agent finds the best action in all states, the system records the results (the number of steps used to train the agent).

Table 4.3 shows the means and standard deviation of steps of five methods in four environments without noise. Both binary feedback and ranking feedback methods exhibit superior performance, achieving speeds around ten times faster than SARSA in each environment. Particularly, in the $10 * 10$ grid-world, binary feedback, and ranking feedback methods require only about $2,000$ steps to learn the best policy, while SARSA needs $30,000$ steps. Moreover, these two methods demonstrate smaller standard deviations than other methods, indicating more stable performance. Figure 4.8 displays the learning curves of 0% noise experiments, which also reveal that binary feedback and ranking feedback methods are comparable. Our hypotheses tests using Mann-Whitney U test with Bonferroni Correction indicate that the p-values between the results of binary feedback and ranking feedback methods are always greater than 0.0125 (0.05/4), signifying no significant difference between them.

Moreover, the state value feedback method, though relatively inferior, still achieves rapid learning of the optimal policy in most cases. As for scaling feedback, it performs the worst among the four IRL methods but still outperforms the SARSA method. In addition to ranking feedback, the results of hypotheses tests between binary feedback and the other three benchmarks show that the p-values are all less than 0.0125, indicating statistical significance.

Moreover, Table 4.4 shows the results in experiments with 5% noise. Overall, binary feedback exhibits the best performance, consistently finding the best policy fastest in all

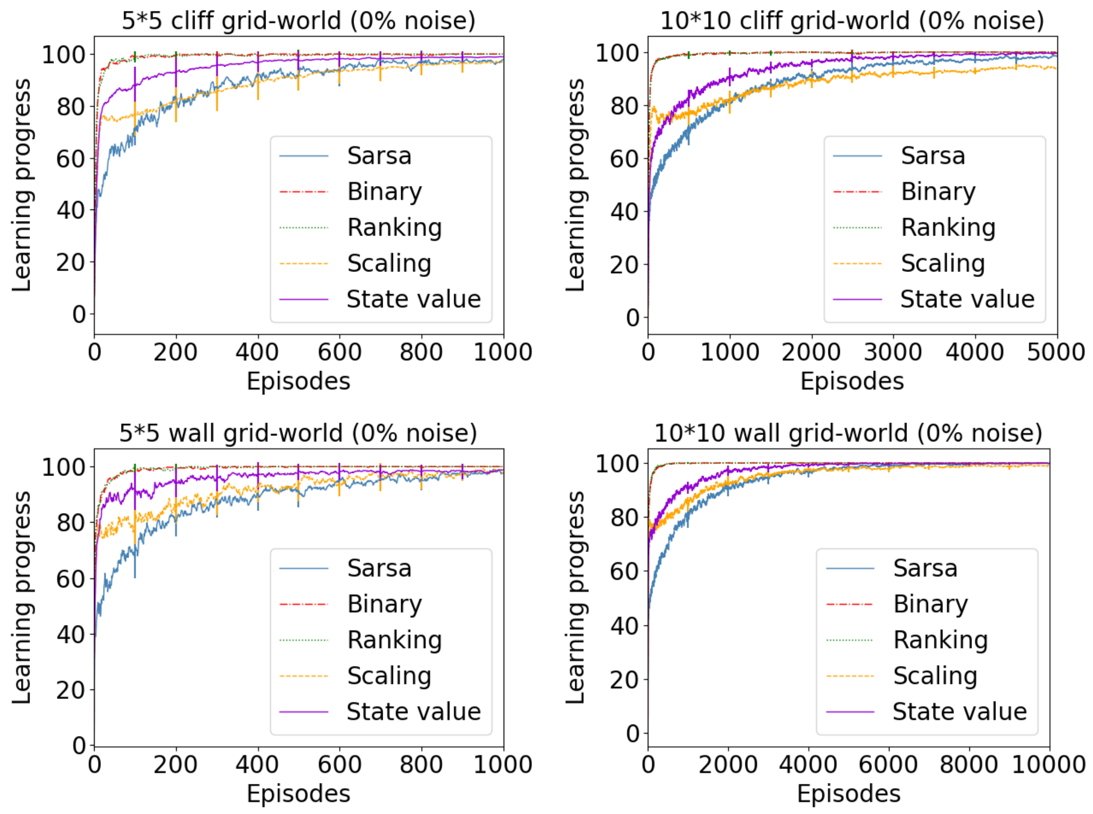


Figure 4.8 The X-axis represents the training episodes. The y-axis represents the learning progress, which is the percentage of states that learned the best solution. Since all the results are relatively stable, the standard error is difficult to visually distinguish the difference between the different methods. Therefore, in order to reflect the stability of different methods, we use standard deviation in our figure. The vertical line on the curve represents the standard deviation of the learning progress at that point.

environments. This indicates that binary feedback demonstrates greater robustness to the noise than other benchmarks. In addition, the ranking feedback method performs slightly worse than binary feedback. However, it is the only method, apart from binary feedback, capable of finding the optimal policy consistently even under the influence of noise.

On the other hand, both scaling feedback and state value feedback methods perform unsatisfactorily. Among them, the state value feedback method struggles to find the optimal policy under the influence of noise. In Figure 4.9, the state value feedback method only manages to learn around 80% of the correct policy. Similarly, the scaling feedback method fails to find the optimal strategy in the cliff grid-world. In more detail, the presence of cliff traps in the cliff grid-world can directly lead to the death of the agent, making wrong feedback in key states potentially fatal and hindering the agent's ability to learn the correct strategy. Furthermore, hypotheses tests are conducted between binary feedback and other benchmarks, and the test results show that all p-values are less than 0.0125, indicating significant differences between binary feedback and the other methods.

Table 4.4 The table shows the form of reward signal experimental results in four different grid-world with 5% noise of human feedback. The results are the means and standard deviation of steps used to find the best solution in 100 experiments. "None" indicates that the agent cannot find the best solution. The red results mean that it has the best result among all methods in an environment.

methods	5*5 cliff	10*10 cliff	5*5 wall	10*10 wall
SARSA	2153.35 +/- 328.95	43641.6 +/- 4420.46	2475.35 +/- 460.33	31497.15 +/- 3659.7
Binary	781.4 +/- 185.46	7040.85 +/- 2308.02	350.5 +/- 127.51	4360.85 +/- 1684.01
Ranking	1036.9 +/- 209.80	11788.25 +/- 4778.40	480.35 +/- 204.72	6437.55 +/- 2757.17
Scaling	None	None	1592.05 +/- 497.04	28235.55 +/- 8703.68
State Value	None	None	None	None

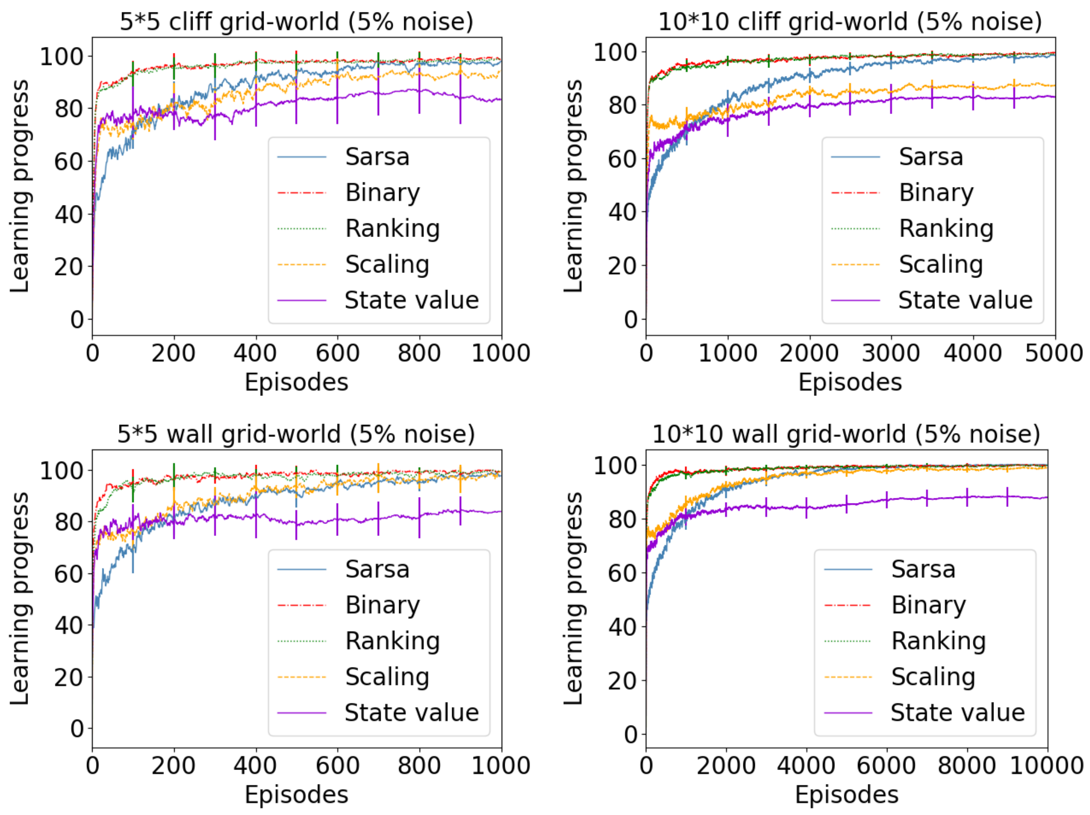


Figure 4.9 The X-axis represents the training episodes. The y-axis represents the learning progress, which is the percentage of states that learned the best solution. For the details see the caption of Figure 4.8

In summary, binary feedback outperforms scaling feedback, ranking feedback, and state value feedback. First, it imposes the smallest cognitive burden, as it only requires distinguishing between "best actions" and "other actions". In contrast, other methods, such as ranking feedback, demand increased cognitive effort from humans to provide more detailed reward signals. Additionally, binary feedback exhibits superior performance in both noisy and noiseless experiments, indicating that providing more feedback details does not necessarily improve performance.

4.2.3 MAIRL Evaluation

In this set of experiments, we evaluate the performance of MAIRL and our review model (see Section 4.1.3) using the $10 * 10$ cliff grid-world task (see Section 4.2.1). Specifically, we set up two additional multi-advisor IRL methods to compare with our MAIRL system. The first is MAIRL-no review. It does not use the review model and only provides feedback once for each state-action pair and remembers the past rewards. The previous reward is used for the same state-action pair. The second is MAIRL-unlimited, which indicates that the agent obtains feedback after each action but without recording the historical advice. Table 4.5 shows the setting of MAIRL experiments. For each experiment, there are five advisors, and the trustworthiness of advisors is sampled from ERGd. The means of advisors' trustworthiness are set in $\{0.51, 0.52, ..., 1\}$, and the standard deviation of advisors' trustworthiness is set to 0.2.

Table 4.5 The max episodes are 500. Max number of actions in each episode is 200. Each experiment runs 100 times. There are 5 advisors.

description	value
max episodes	500
max actions	200
number of advisors	5
trustworthiness means of advisors	0.51, 0.52...1
trustworthiness standard deviations of advisors	0.2
experimental times	100

In addition, the maximum number of episodes of the 10*10 cliff grid-world is set to 500. This setting is based on the observation that the IRL method typically completes training in approximately 150 episodes without noise interference. Also, we scale up the maximum number of episodes considering the effect of noise. Moreover, the agent easily goes into a loop because of the noise, so the agent can perform at most 200 actions in each episode to save computational power costs. If the agent finds the best solutions, it stops training and records the results. Moreover, each set of experiments needs to be repeated 100 times. This repetition helps in obtaining statistically meaningful results and validating the stability of the MAIRL system's performance.

Figure 4.10 shows the results of MAIRL experiments. Overall, the results demonstrate that the policy trained by MAIRL with the review model exhibits superior performance, being closer to the optimal policy compared to MAIRL without a review model. Across

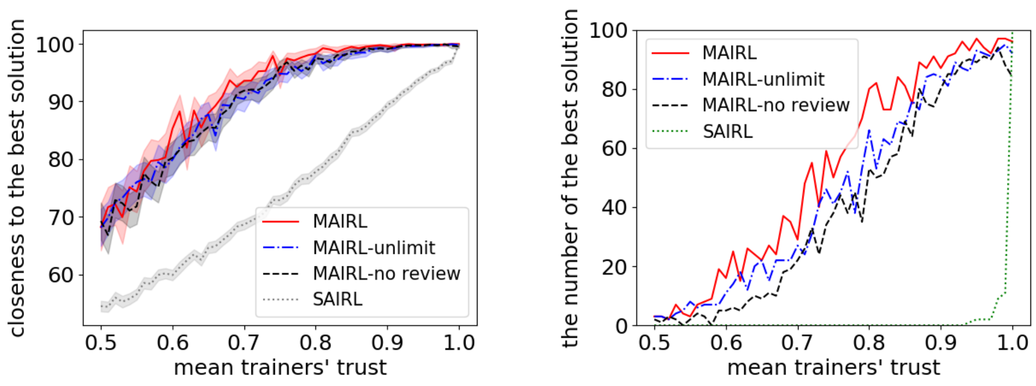


Figure 4.10 The left figure shows the closeness to the best solution as the advisors' trust increases in different methods. The X-axis represents the mean of advisors' trust from 0.51 to 1. The Y-axis represents the average closeness to the best solution at the end of training in 100 experiments. There are 50 points in each curve. Each point is the mean of 100 experimental results. The part with a transparent color is the 95% confidence interval error bar. The right figure shows the number of best solutions in 100 experiments, as the advisors' trust increases. The X-axis represents the mean of advisors' trust from 0.51 to 1. The Y-axis represents the number of the best solution of different methods in 100 experiments. There are 50 points in each curve.

a range of advisors' trustworthiness mean values from 0.55 to 0.9, MAIRL with the review model outperforms both MAIRL-unlimit and MAIRL-no review methods. However, for mean values below 0.55 and above 0.9, three MAIRL methods exhibit similar performance.

Out of 100 Mann-Whitney hypotheses tests conducted between MAIRL and the other methods without a review model, 38 results show $p < 0.0166$ $(0.05/3)$. This suggests that MAIRL performs significantly better than the other methods in those cases. While MAIRL-unlimit has the advantage of providing unlimited feedback, indiscriminate feedback may lead to incorrect revisions of the correct feedback. As a result, its feedback maintains a fixed accuracy rate, limiting its performance. This observation highlights the importance of having a review model and relevant evidence to improve the efficiency of agent learning in MAIRL.

Moreover, the review model utilizes a small amount of additional feedback to improve the agent's learning ability. Figure 4.11 (left) illustrates that MAIRL-unlimit has the highest number of feedback instances, approximately ten times more than the other two methods, MAIRL and MAIRL-no review. Both MAIRL and MAIRL-no review exhibit relatively lower feedback costs, with MAIRL slightly higher than MAIRL-no review. As depicted in the figure, the amount of feedback from MAIRL decreases as the mean advisors' trust increases. This is related to the real trust of advisors. In more detail, when the trustworthiness of advisors is relatively low, the confidence value i_t becomes very low, resulting in a higher probability of the review process taking place.

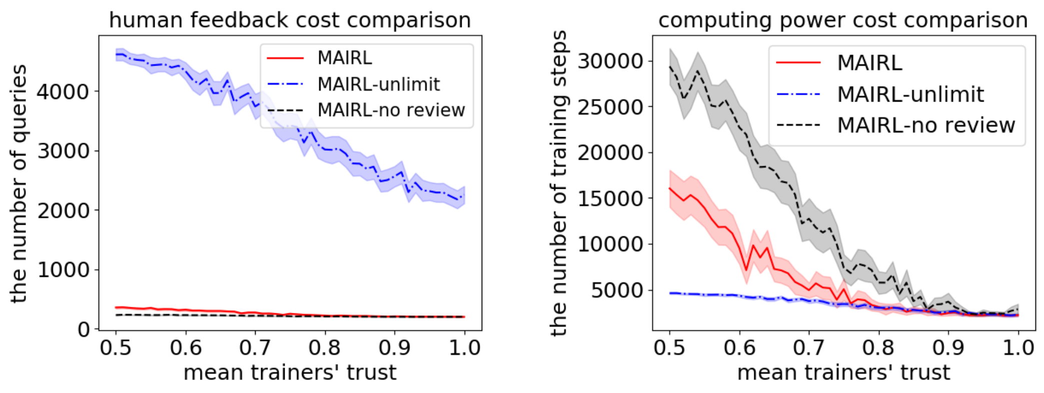


Figure 4.11 The left figure shows the average number of queries in 100 experiments as the advisors' trust means increases. The X-axis represents the mean of advisors' trust from 0.51 to 1. The Y-axis represents the average number of queries in 100 experiments. There are 50 points in each curve. Each point is the mean of 100 experimental results. The transparent color area is the 95% confidence interval error bar. The right figure shows the average number of training steps in 100 experiments as the advisors' trust means increases. Different from the left figure, The Y-axis represents the average number of training steps in 100 experiments.

Furthermore, the review model reduces the risk of trapping the agent in a loop. Figure 4.11 (right) shows the average number of training steps for the three MAIRL methods in 100 experiments under varying means of advisors' trustworthiness. MAIRL-no review exhibits the highest average training steps. When the mean is less than 0.8, the average training steps of MAIRL-no review are almost twice that of MAIRL. This indicates that if we only trust the rewards of the first feedback of each state-action pair, then the agent easily falls into a loop because of incorrect rewards. On the other hand, MAIRL-unlimit requires the fewest steps, but it comes at a significant human cost. This is because if the previous reward puts the agent in the loop, then the infinite queries have a big likelihood of changing their reward, which can make the agent break out of the loop. Similarly, MAIRL, with the assistance of the review model, can also break the loop by potentially asking advisors for additional feedback, leading to a modification of incorrect advice and enabling the agent to escape the loop.

Finally, MAIRL has a more powerful performance compared to single advisor IRL. In Figure 4.10, experimental results show that when the average trust of advisors is 0.7, the agent can learn the best solutions of 90%, while the single-advisor IRL can only learn around 70%. Before the mean advisor's trust was 95%, it was almost difficult for the single-advisor IRL method to learn all the best solutions, but MAIRL can learn the best solutions many times.

4.3 Summary

In this chapter, we finish the Objective **O2.1** and **O2.2** in Section 1.3.2. We use our MAB-SDM (see Chapter 2) to propose the MAIRL method, which is an IRL system that can combine the advice of multiple advisors. It can aggregate a set of feedback from non-perfect advisors into a more reliable reward for RL agent training in a reward-sparse environment. Moreover, we conduct grid-world experiments to evaluate MAIRL. The results of feedback form experiments show that binary feedback outperforms ranking feedback, scaling feedback, and state value feedback in terms of training performance and the robustness of incorrect feedback. Furthermore, the results of the MAIRL experiments show that the policy trained by the MAIRL is closer to the optimal policy than that without a review model.

Chapter 5

Sequential Binary Decision-Making by Maximizing the Utility

In this chapter, we adopt the MABSDM method for a utility-maximizing problem in binary decision-making scenarios. Specifically, based on the multi-advisor binary sequential decision-making problem, we consider the value of problems and the cost of querying advisors in sequential decision-making. In this setting, the challenge is selecting a subset of advisors by balancing the value of each problem and the cost of querying advisors, to make sequential optimal decisions (see Objective **O3.1**, and **O3.2**). To complete this objective, we propose a novel strategy, Multi-Advisor Dynamic Decision-Making (MADDM), for optimally selecting a set of advisers in a sequential binary decision-making setting, where multiple decisions need to be made over time. Specifically, our approach considers how to simultaneously (1) select advisors by balancing the advisors' costs and the value of problems, (2) learn the trustworthiness of advisers dynamically without prior information by asking multiple advisers, and (3) make optimal decisions without access to the ground truth, improving this over time.

In addition, we conduct extensive experiments that compare MADDM to a variety of methods that combine state-of-the-art approaches, including budget-limited decision-making, ϵ-greedy selection, and the EM method, and we benchmark performance against the optimal utility that could be gained with perfect knowledge. The results show that MADDM outperforms the other two methods in almost all environments.

The rest of the chapter is structured as follows. Firstly, we describe the MADDM method in Section 5.1. Secondly, we present the experiments and results in Section 5.2. Lastly, we summarize our work in Section 5.3.

5.1 Model Description

In practice, a successful decision often comes with a reward, while a failed decision and querying advisors have a cost. An expensive decision, such as an investment of 10 million pounds, requires querying multiple experts to make a decision. On the other hand, mundane decision, such as what to have for dinner, does not need external advice. The challenge is querying a suitable set of advisors for decision-making by balancing the value of the problem and the cost of querying advisors.

In this chapter, we propose MADDM to solve this problem. Specifically, the design of MADDM consists of four interdependent components. The first is the advisor selection model, which assigns a set of advisors to each problem by balancing the value of the problem and the cost and trustworthiness of advisors. The challenge is utilizing the present evidence of advisors' trustworthiness to select an optimal set of advisors based on the query cost of advisors and the value of the problem. In more detail, the advisor selection model selects advisors one by one by calculating the marginal utility of each advisor until increasing utility cannot be obtained if continue to select more advisors (see Section 5.1.2). The second component is our BCT trustworthiness model (see Section 3.1.2), which can be used as a weight in the decision model and to calculate the contributions of advisors in the advisor selection model. The third is our BBWVE decision model (see Section 3.1.3), which makes a decision after receiving the advisors' opinions. The last is a review update model. This model can improve the accuracy of advisors' trustworthiness by reviewing historical advice. Figure 5.1 provides a graphical overview of the structure of MADDM.

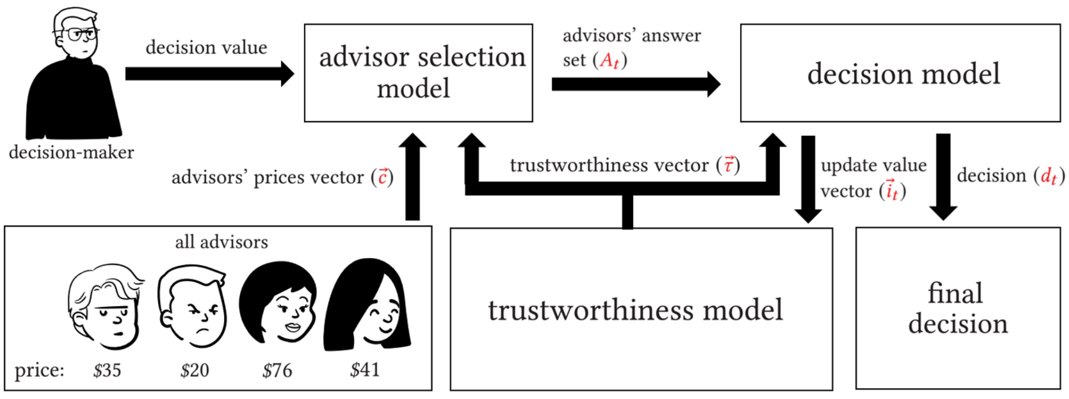


Figure 5.1 The advisor selection model can select a subset of advisors from all advisors by considering the value and risk of problems, the advisors' cost, and trustworthiness. The decision model uses advisors' trustworthiness and the advice set to make the decision and the estimated evidence for updating the trustworthiness. The trustworthiness model builds and updates the advisors' trustworthiness.

5.1.1 Problem Formalization

Given a problem t, $v_t^+ \in \mathbb{R}$ is the value that the decision-maker obtains if the decision is correct. We denote with $v_t^- \in \mathbb{R}$ a cost of a wrong decision. Therefore, the value of the problem is represented by the tuple $v_t^\pm = (v_t^+, v_t^-)$. Moreover, since we rely on advisors' advice to make decisions, we need to incentivize them by introducing a payment system. For each advisor $x \in X$, c_x is the fixed price of each query. In addition, we denote with $\vec{c}$ the vector containing all the advisors' prices.

Therefore, we describe any possible selection through a function f_s that, to every tuple $I := (t, \vec{\tau}, v_t^\pm, \vec{c}) \in T \times [0,1]^{|X|} \times [0, +\infty]^2 \times [0, +\infty]^{|X|}$, associates a subset of advisors $Y_t \in \mathcal{P}(X)$, where $|X|$ is the cardinality of X and $\mathcal{P}(X)$ is the power set of X; we call f_s the *selection function*, so we have $Y_t = f_s(I)$.

Moreover, we recall that $A_t = \{A_{Y_t}^{pos}, A_{Y_t}^{neg}\}$ denotes the advice set of problem t and d_t denotes the decision of problem t. Let $u_t(f_s(I))$ denote the utility of the decision t to the decision-maker. Accordingly, for each decision, t, the total cost to the decision-maker to query the advisors in $f_s(I)$ is $C_t(f_s(I)) = \sum_{x \in Y_t} c_x$. Then we have:

$$u_t\left(f_s(I)\right) = \begin{cases} v_t^+ - C_t\left(f_s(I)\right) \text{ if } d_t = d_t^*, \\ v_t^- - C_t\left(f_s(I)\right) \text{ otherwise }. \end{cases} \tag{5.1}$$

In particular, the sum of the utilities for all the decisions is $u(f_s(I)) = \sum_{t \in T} u_t(f_s(I))$. Since each advisor has a different cost, the final utility depends on the advisor selection function adopted. In this framework, the goal of the decision-maker is to find the selection function f_s to maximize its utility.

5.1.2 Advisor Selection

The overall aim of the system is to maximize utility, which requires balancing the trade-off between the cost of advisors and the value of problems. Typically, the costs of asking all advisers may exceed the problem value, even if the decision is correct, so it is rarely optimal. For example, for a problem with a value of \$10, it is not worth spending \$100 to query advisors.

Our method selects the set of advisors according to the value of the problem and estimates their contributions to a decision. We assume their trustworthiness is initially unknown, and therefore all advisers have equal trustworthiness. This knowledge is updated over time but is not reliable at first. Therefore, focusing too early on seemingly good advisors can lead to sub-optimal decisions. To address this, our system solves a multi-armed bandit problem in which it has to balance the exploration of new advisors with the exploitation of the knowledge it has already gathered. Among many possible algorithms used to solve the multi-armed bandit problem, we use Thompson Sampling (see Section 2.4.1), which samples a probability from a Beta distribution to compute the marginal contribution of each advisor. This sampled probability encapsulates both the advisor's trustworthiness (exploitation) and potential (exploration). In addition, this sampled probability serves as an input for calculating the advisor's marginal contribution, which is quantified as a value for the model to determine whether selecting the advisor is worthwhile (see more details in the following). In contrast, although other multi-armed bandit approaches like UCB and ϵ-greedy can balance exploration and exploitation when selecting advisors, it is difficult to quantify the marginal contribution.

In Algorithm 3, we sketch the pseudo-code of our selection function f_s. We recall that $\vec{\tau}$ denotes the trustworthiness vector that contains the trustworthiness of each advisor, and $\vec{c}$ are their costs. Let $\vec{\alpha}$ and $\vec{\beta}$ denote the estimated evidence vectors that contain the evidence of all advisors. Given a problem $t \in T$, let P_t^{pos} and P_t^{neg} denote the probability that $d_t^* = pos$ and $d_t^* = neg$, respectively (see Section 3.1.3). We denote with U_t the vector containing the advisors' utilities.

Algorithm 3 Pseudo-code of the Advisor Selection algorithm

```
1: **Input**: $t, \vec{\tau}, \vec{\alpha}, \vec{\beta}, v_t^+, v_t^-, \vec{c}$
2: $initialize\ P_t^{pos}, P_t^{neg}, U_t = A_{Y_t}^{pos} = A_{Y_t}^{neg} = \varnothing$
3: **while** $true$ **do**
4:     **for** advisor $x$ in $X$ **do**
5:         $\tau_x' \leftarrow ThompsonSampling(\alpha_t^x + 1, \beta_t^x + 1)$
6:         $u_t^x \leftarrow UtilityComputation(\tau_x', v_t^+, v_t^-, P_t^{pos}, P_t^{neg}, c_x)$
7:         $U_t.append(u_t^x)$
8:     $u_t^{x^*} = Max(U_t)$
9:     **if** $u_t^{x^*} > 0$ **do**
10:        **if** $d_t^{x^*} = pos$ **do**
11:            $A_{Y_t}^{pos}.append(x^*)$
12:        **if** $d_t^{x^*} = neg$ **do**
13:            $A_{Y_t}^{neg}.append(x^*)$
14:    $P_t^{pos}, P_t^{neg} \leftarrow DecisionModel(A_{Y_t}^{pos}, A_{Y_t}^{neg}, \vec{\tau})$
15:    $U_t = \varnothing$
16:    $X.remove(x^*)$
17: **until** $u_t^{x^*} \leq 0$
18: **Output**: $A_{Y_t}^{pos}, A_{Y_t}^{neg}$
```

In more detail, after initializing the advice probabilities P_t^{pos} and P_t^{neg}, the advice sets $A_{Y_t}^{pos}$ and $A_{Y_t}^{neg}$, the utility vector U_t, and the trustworthiness vector (Line 2), the model enters a loop for selecting advisors (Line 3). Let V_t^x, u_t^x denote the expected contribution and the marginal utility of the advisor x in problem t (the calculation method of V_t^x will be introduced in the following). Recall that c_x is the price of advisor x. Their relationship can be expressed as follows:

$$u_t^x = V_t^x - c_x. \tag{5.2}$$

In each round of advisor selection, we need to compute the marginal utility u_t^x of each advisor and select the advisor x with the most utility, i.e., $x^* = \arg\max_{x \in X} u_t^x$, which is our estimation of the advisor that maximizes the expected profit for the decision-maker (Lines 4-8).

Computing the marginal utility u_t^x is achieved in two steps. First, for each advisor, x, we define a Beta distribution $\mathrm{Beta}(\alpha_t^x + 1, \beta_t^x + 1)$ and sample from it to obtain the Beta trustworthiness τ_x'. We only use it to compute the utility u_t^x of the advisor x (Line 5), whereas the model does not use τ_x' for real decision-making. When there is little evidence regarding an advisor, e.g. when $\alpha = 1$ and $\beta = 1$, the Beta distribution has a large variance, which can increase the risk of making incorrect decisions.

Second, we need to know the contribution V_t^x of each advisor x. Let us now assume that advisor x gives *pos* to a problem t; the case in which the advisor advises *neg* follows a similar routine. In order to compute its contribution, we first add x to the set $A_{Y_t}^{pos}$ and proceed to calculate the probabilities $P_t^{pos'}$ and $P_t^{neg'}$ by the BBWVE method (see in Section 3.1.3). The value $P_t^{pos'}$ and $P_t^{neg'}$ describes the probability that $d_t^* = pos$ and $d_t^* = neg$, respectively. Therefore, the wider the gap between P_t^{pos} (the probability without the advice of advisor x) and $P_t^{pos'}$, the larger the advisor's contribution. Let $P_{pos} := P(d_t^* = pos)$ and $P_{neg} := P(d_t^* = neg)$ denote the *a priori* probability that the advice is positive or negative, respectively. In addition, let $\Delta V_{t,pos}^x$ denote the marginal value if the new advisor x selects the positive option. It can be expressed as:

$$\Delta V_{t,pos}^x = P_{pos} |P_t^{pos'} - P_t^{pos}| * (v_t^+ + v_t^-). \tag{5.3}$$

where the value $|P_t^{pos'} - P_t^{pos}|$ represents the change of the advice probability if advisor x participates in the decision. In addition, the values v_t^+ and v_t^- are utilized to calculate the marginal value of making a correct decision and an incorrect decision, respectively. Similarly, let $\Delta V_{t,neg}^x$ denote the marginal value if the new advisor x selects the negative option. It can be expressed as:

$$\Delta V_{t,neg}^x = P_{neg} |P_t^{neg'} - P_t^{neg}| * (v_t^+ + v_t^-). \tag{5.4}$$

After we compute $\Delta V_{t,pos}^x$ and $\Delta V_{t,neg}^x$, we compute the expected contribution V_t^x as:

$$V_t^x = (\tau_x' - (1 - \tau_x')) * (\Delta V_{t,pos}^x + \Delta V_{t,neg}^x). \tag{5.5}$$

where τ_x' indicates the probability that advisor x providing correct advice, while $1 - \tau_x'$ is the probability that advisor x providing an incorrect advice. Finally, the algorithm computes the utility u_t^x by Equation 5.2.

If $u_t^{x^*} > 0$, the advisor x^* is selected, which represents that the contribution is greater than his or her cost. The selected advisor x^* needs to provide the advice for problem

t. Depending on the advice of the advisor x^*, he or she can be added to $A_{Y_t}^{pos}$ or $A_{Y_t}^{neg}$ (Lines 9-13), which is used to update the advice probability P_t^{pos} and P_t^{neg} (Line 14). After each selection, we need to recalculate the marginal utility of each advisor for selecting the next advisor because their marginal utilities change. For example, there are three advisors $\{A, B, C\}$. If we select advisor A, we can obtain the advice of advisor A. Then, the probability P_t^{pos} and P_t^{neg} changes. Therefore, to select the next advisor, we need to recalculate the marginal contributions of advisors B and C based on new P_t^{pos} and P_t^{neg}. The model repeats Lines 4-16 to select advisors one by one until $u_t^{x^*} \leq 0$ (Line 17), and outputs the final advice set $(A_{Y_t}^{pos}, A_{Y_t}^{neg})$ (Line 18).

5.1.3 Review Update

MADDM is an online problem without access to ground truth. Moreover, the reliability of initial trustworthiness is low. Therefore, the update of the trustworthiness $\vec{\tau}$ relies on the evidence from new decisions. And the decisions, in turn, rely on the trustworthiness $\vec{\tau}$. This dynamic loop is used for building the model to make the trustworthiness and the aggregating advice more accurate. Therefore, similar to the EM method, after each decision, we continuously update the trustworthiness of the advisors through advice on past decisions.

Algorithm 4 describes how the review update works. Let $\vec{A}_{past}^{pos}$, $\vec{A}_{past}^{neg}$ denote the vector that contains the past advice set, and we recall that $\vec{\tau}$ denote the trustworthiness vector that contains all advisors' trustworthiness. Let $\vec{\tau}_0$ denote the old trustworthiness vector, and $\Delta\tau$ denotes the sum of the difference between the old trustworthiness vector $\vec{\tau}_0$ and the new trustworthiness vector $\vec{\tau}$. In addition, based on the present P_t^{pos} P_t^{neg}, the trustworthiness can be updated (by Equation 3.4 and 3.5) (Line 7). Before updating trustworthiness, the last update of the problem t needs to be removed. Furthermore, let V_s denote the threshold of $\Delta\tau$ for terminating the update. V_s usually is set to a small value. In addition, we note that $\Delta\tau$ is used to judge the update step size of $\vec{\tau}$. Specifically, when $\Delta\tau$ is smaller than V_s, the model stops updating (Line 9).

Algorithm 4 Pseudo-code of the review maximization algorithm

```
1:  **Input**: $\vec{A}_{past}^{pos}, \vec{A}_{past}^{neg}, \vec{\tau}, V_s$
2:  *initialize* $\Delta\tau = 0, \vec{\tau}_0 = \vec{\tau}$
3:  **while** *true* **do**
4:      **for** $A_{Y_t}^{pos}, A_{Y_t}^{neg}$ in $\vec{A}_{past}^{pos}, \vec{A}_{past}^{neg}$ **do**
5:          $P_t^{pos}, P_t^{neg} \leftarrow f(A_{Y_t}^{pos}, A_{Y_t}^{neg}, \vec{\tau})$
6:          $\vec{\tau}_0 = \vec{\tau}$
7:          $\vec{\tau} \leftarrow TrustworthinessUpdate(P_t^{pos}, P_t^{neg})$
8:          $\Delta\tau = sum(\vec{\tau} - \vec{\tau}_0)$
9:  **until** $\Delta\tau \leq V_s$
10: **Output**: $\vec{\tau}$
```

5.2 Experiments

In this section, we present the problem-advice experiments to evaluate our method. Specifically, we compare our method with two cost-constraint-based methods. The first is the Fixed Number of Advisors method (FNA), which means that the decision-maker selects a fixed number of advisors for giving advice to each decision. The second is the Budget-Constraint method (BC), which represents that there is a budget constraint to stop selecting advisors for each problem. For both approaches, we combine these with different advisor-selection criteria.

5.2.1 Setting

To the best of our knowledge, there is no standard environment to run decision experiments. For this reason, we rely on synthetically generated ones. In more detail, the environment we generate includes $1,000$ problems with binary options and different values. The full set of advisors consists of 30 simulated advisors with different advice accuracy and costs. Similarly, we employ the ERGd method (see Section 3.1.3.2) to sample both the profits and losses of each problem. Specifically, during the experiments, the decision-maker selects a set of advisors to enquire and make decisions using different methods. After making decisions on $1,000$ problems, the decision-maker obtains the final utility. Due to the probabilistic nature of the experiments, each experiment is repeated for 100 different runs to obtain statistically significant results. To reduce variance and bias, all methods are run using the same conditions. That is, although the conditions vary between runs, the same set of runs is used to compare the methods (i.e., using the same set of advisor qualities and prices, the same problem sequence, and the same profits and losses of problems).

We consider different ratios between the decision's value and the advisor's cost, which leads us to define two sets of experiments. In the first set, both the problem profits and losses are sampled from an ERGd whose mean and standard deviation are equal to 100. In the second one, the mean and the standard deviation of the ERGd are both changed to 500. Due to the large deviation, the problem values are highly volatile. Hence, some decisions may be worth more than 1000, and some may be worthless but the lower limit is 0.

Furthermore, the real accuracy of advisor x, i.e. τ_x^r, is sampled from an ERGd whose standard deviation σ is fixed at 0.3 while its mean μ ranges in the set $\{0.5 + 0.01 * k\}$ where $k = \{1, 2, \ldots, 50\}$. For example, if τ_x^r is 0.8, the advisor x has 80% probability of giving correct advice. Hence, we consider 50 different frameworks in which the average trustworthiness increases every time. Finally, we assume that the cost of each advisor is proportional to its real trustworthiness. In practice, higher quality often comes at a higher cost. For example, senior advisors are more costly than junior ones. Similarly, more advanced machine learning algorithms typically require higher computational costs. However, this is only a correlation and not always the case for each instance. To achieve this correlation, the cost of each advisor is sampled from an ERGd whose average is $\tau_x^r * 20$ and whose standard deviation is 10. Note that the correlation makes the problem more challenging since the system has to make trade-offs between cost and quality. Without such a correlation, there is a high likelihood of a cheap and reliable advisor which makes the problem easier to solve but also less realistic.

Table 5.1 MADDM experiment setting

setting	value
env1: problem profits v_t^+ $mean, std$	$100, 100$
env1: problem loss v_t^- $mean, std$	$100, 100$
env2: problem profits v_t^+ $mean, std$	$500, 500$
env2: problem loss v_t^- $mean, std$	$500, 500$
advisor cost c_x $mean, std$	0 to 20,10
real trustworthiness $mean, std$	from 0.51 to 1, 0.3

We used three different exploration methods, they are UCB, Thompson Sampling, and ϵ-greedy (see Section 2.4.1), and two rules of the advisor selection (trustworthiness, cost-effectiveness) to combine with FNA and BC, respectively. The aggregation method of FNA and BC is EM, which can maximize the sample utilization and has been verified multiple times in truth inference (Demartini et al., 2012; Gemalmaz and Yin, 2021).

In more detail, in terms of advisor selection strategies, UCB, Thompson Sampling, and ϵ-greedy are effective for solving the multi-armed bandit problem. We experimented with a range of values and found that the ϵ-greedy method has the best performance when $\epsilon = 0.1$ (we also tested ϵ = 0.05, 0.15, 0.2, 0.25) for all methods.

The criteria for advisor selection contain trustworthiness and cost-effectiveness. For example, if trustworthiness is the rule, the greedy strategy always selects the advisor with greater trustworthiness but ignores their cost. Cost-effectiveness is a method we improved from work (Xia et al., 2015). The cost-effectiveness of the advisor x can be expressed by $c_x/(\tau_x - 0.5)$, which means how much cost is the improvement of trustworthiness for advisor x. Our results show that it has a better performance than trustworthiness.

For FNA and BC, we also test their performance under different hyper-parameters. First, we test the performance of FNA by setting the number of advisors from 1 to 10. The results show that five advisors have the best performance. Second, BC, we use 5%, 10%, 15%, 20%, 25% of the value (profit + loss) of each problem as the budget constraint, and 10% has the best performance.

Table 5.2 The meaning of the abbreviations is env1 (SD): environment 1 and standard methods, env2 (SD): environment 2 and standard methods env1 (EF): environment 1 and all methods with exploration-first model, env2 (EF): environment 1 and all methods with exploration-first model.

	MADDM	FNA	BC	RV
env1 (SD)	5.19 ± 7.68	3.76±5.95	2.85±7.51	3.27±2.84
env2 (SD)	42.69 ± 27.38	30.89±33.03	35.46±28.11	34.16±16.31
env1 (EF)	7.09 ± 4.43	5.15±4.32	6.26±4.79	3.24±2.93
env2 (EF)	44.97 ± 22.84	38.89±24.11	37.58±22.68	34.14±16.20

To clearly understand the performance of our method, FNA, and BC, we selected two other methods for comparison. The first is random voting (RV). It randomly selects three advisors and combines them by majority voting. Another one is the best utility (BU). It describes the maximum utility the decision-maker can obtain, which means all the decisions are correct, and the advisor cost is 0.

In addition, the method with the trustworthiness model is easily misled by malicious advisors when the mean advisors' accuracy is low. In practical applications, the methods for solving the problem include adding some decisions with ground truth, selecting several advisors with high accuracy to participate in decision-making, or considering the prior information of advisors. In this chapter, our assumptions are no ground truth and no prior information, so we design the exploration-first model to solve this problem. In the first few decisions, the model selects all advisors to give advice on problems to increase the accuracy and then back to the method's standard advisor selection strategy. We use this model before rounds $1 - 15$, respectively, and the results show that the three methods perform best when the model is used before the 10 round. Therefore, we added the exploration-first model to our method, FNA, and BC, and did additional experiments in two environments.

5.2.2 Results

Table 5.2 shows the mean and standard deviation of the utility in each environment. Overall, our MADDM method has the best performance in terms of the average utility in almost all environments. In all the experiments, the average utilities obtained by the exploration-first methods are significantly bigger than the others. Moreover, the standard deviation of the utilities is also reduced, which means that the result is more consistent. We did 600 ($3 * 50 * 4$) pairs of Mann-Whitney Tests with Bonferroni Correction between MADDM and FNA, BC, and Random Voting (RV) with 50 different average advisors' accuracy in four different environments. We observe that 527 out of the 600 results have significant differences ($p < 0.05/3$).

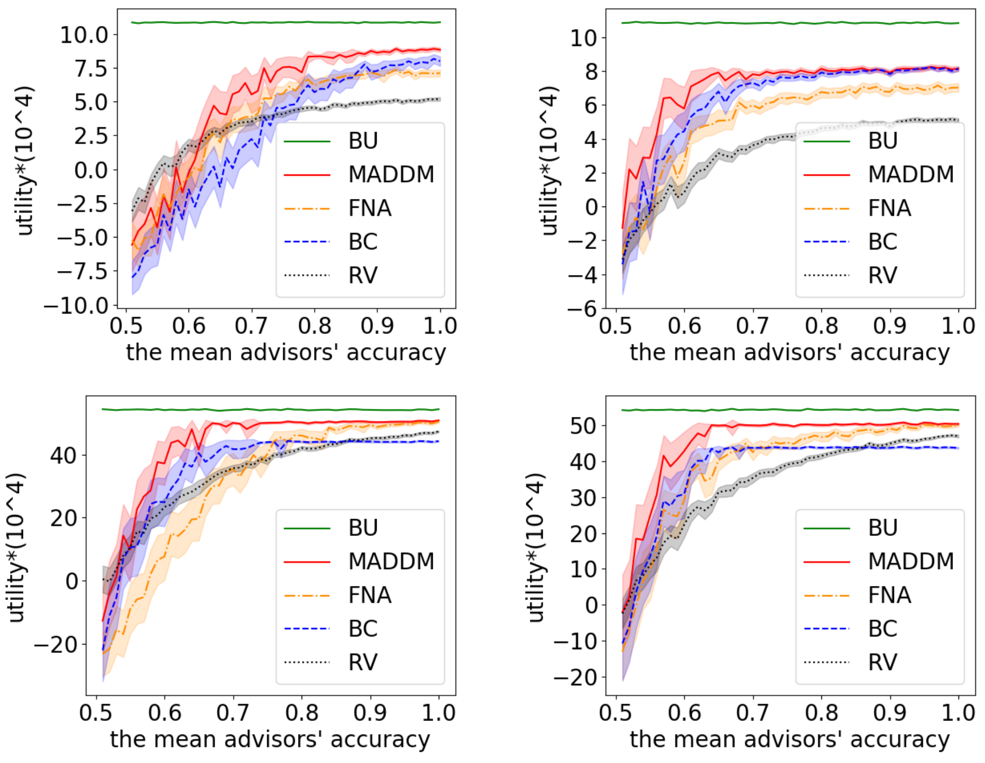


Figure 5.2 In four figures, the X-axis represents the mean advisors' accuracy from 0.51 to 1. The Y-axis represents the average utility of 100 experiments. The half-transparent area, along with the curve, is the 95% confidence interval error bar. Two figures in the first line show the results of Environment 1 (mean, standard deviation = 100, 100). Two figures in the second line show the results of Environment 2 (mean, standard deviation = 500, 500). In addition, the left two figures represent the standard methods, and the right two figures are the exploration-first-based methods. MADDM = multi-advisor dynamic decision-making(ours); FNA = ϵ-greedy fixed number of the advisor EM; BC = ϵ-greedy budget-limited EM; RV = random voting.

Figure 5.2 describes the utility curves of different methods as the advisors' accuracy increases. In the vast majority of cases, MADDM obtains more utility than FNA and BC for all the possible accuracy.

Specifically, MADDM automatically selects the advisors by balancing the advisor's cost and the problem values without any hyper-parameters, which makes MADDM less prone to select an insufficient number of advisors or to waste costs. In the two methods based on cost-effectiveness, they need to set the number of advisors and budget proportion to control the advisor cost. If the prior distribution is unknown, the values of these hyper-parameters are difficult to determine. Furthermore, if the advisor cost is too small, the reliability of the output is not enough. If the cost is too high, it causes a waste of advisor costs. For example, in Figure 5.2, in Environment 2 (two figures in the second line), we observe that FNA does not select enough advisors when the mean advisors' accuracy is less than 0.8, whereas the best performance of BC has a gap with MADDM when the mean advisors' accuracy is higher than 0.65.

In addition, MADDM has stronger robustness to malicious advisors than FNA and BC. In more detail, RV is better than the other three methods when the mean advisors' accuracy is low in the results of the standard methods (Figure 5.2 left). When there is no ground truth and a significant proportion of bias, the methods with the trustworthiness model are easily misled by malicious advisors. Once the trustworthiness model is misled, then malicious users can sabotage future decisions. However, we observe that MADDM is less prone to be sabotaged than FNA and BC. This is due to the fact that MADDM selects more advisors than other methods at the beginning and decreases as trustworthiness is updated. In addition, the BBWVE method can make more accurate decisions than the Bayesian methods when the trustworthiness is unreliable (see Section 3.2.2).

Similarly, we observe that the performances of MADDM are more robust to the malicious advisors when the problem values are bigger. Since the decisions in the environment 2 are more valuable than the ones in the environment 1, MADDM chooses more advisors to make decisions together at the beginning in environment 2, which helps to increase the reliability of the advice. However, this method causes some costs when the real advisors' accuracy is high. For example, if the trustworthiness of all advisors is 100%, the model only needs to select one cheapest advisor. However, we do not know the real distribution of the mean advisors' accuracy before querying, so it is worth using some cost at first to improve the method's expected utility.

5.3 Summary

In this chapter, we complete Objective **O3.1** and **O3.2**. Specifically, we introduce the Multi-Advisor Dynamic Decision-Making method (MADDM), a novel decision-making method by maximizing the utility based on MABSDM. The model takes into account multiple variables, including the profits and losses of problems, advisors' costs, and trustworthiness. In more detail, it consists of three interdependent components: trustworthiness assessment, advisor selection, and decision-making. Trustworthiness assessment builds and maintains models of the trustworthiness of each advisor. For each sequential decision, advisor selection identifies which advisors to consult. This is similar to a multi-armed bandit problem, which requires a balance of exploration and exploitation. We use Thompson Sampling combined with the decision-making model to compute each advisor's expected marginal contribution and select advisers until the marginal contribution is negative. The third component uses the set of advice to make a decision using the BBWVE method.

Moreover, we test our method through problem-advice experiments in a simulated environment. We also introduce two benchmark methods, one using a fixed number of advisors (FNA) and another one using a fixed budget (BC), which are combined with state-of-the-art sampling and aggregating methods. The results show that MADDM significantly outperforms the benchmark methods.

Chapter 6

Sequential Decision-Making among Multiple Options

In this chapter, to address Objective **O4.1** and **O4.2**, we extend the MABSDM method to a general method, called Multi-Advisor Sequential Decision-Making (MASDM). Compared to MABSDM, MASDM can handle decisions among multiple options, not just binary options. In decision-making among multiple options, the challenge is to deal with conflicts of advisors' advice among multiple options reasonably. Similar to MABSDM, MASDM consists of two parts: a decision model and a trustworthiness model. The decision model makes decisions among multiple options using the advice of advisors by the Bayesian Weighted Voting Ensemble (BWVE) method. The trustworthiness model builds and updates advisors' trustworthiness using a Cautious Trustworthiness (CT) model. Different from the BCT model (see Section 3.1.2)), the update strategy of the CT model considers balancing the expected confidence value of multiple options.

In addition, we evaluate and explain our methods through extensive experiments in simulated environments. Moreover, we apply our method to ensemble machine learning using the experiments by the MNIST database. The results show that MASDM has better decision accuracy and the ability to assess trustworthiness compared to five benchmarks that use state-of-the-art methods, in particular achieving a maximum improvement of 22% in accuracy compared to the Bayesian aggregation methods.

The rest of the chapter is structured as follows. First, we describe the MASDM method in Section 6.1. Second, we present the experiments and results in Section 6.2. Lastly, we summarize our work in Section 6.3.

6.1 Model Description

We now introduce our method called MASDM. Similar to MABSDM, it is an online method composed of a trustworthiness model and a decision model. In terms of the decision model, the difference from the BBWVE method is that the BWVE method calculates the correct probability distribution among multiple options, not binary options. For the trustworthiness model, compared to the BCT model, the CT model considers the expected confidence value, not the probability difference between the decision and other options. Figure 6.1 provides an overview of how MASDM works. For each problem, first, the decision model utilizes the advice set and the trustworthiness vector from the trustworthiness model to make decisions. Second, the decision model calculates the new evidence (introduced in Section 6.1.2) for updating the advisors' trustworthiness.

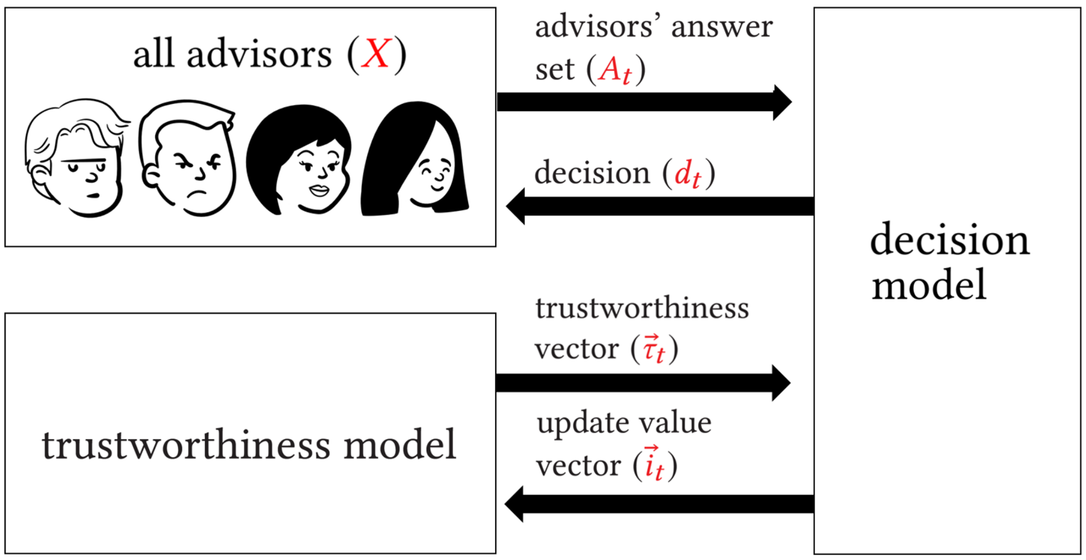

all advisors (X)
advisors' answer set (A_t)
decision (d_t)
decision model
trustworthiness vector ($\vec{\tau}_t$)
update value vector ($\vec{i}_t$)
trustworthiness model

Figure 6.1 For each problem t, a subset of advisor set X provides a set of advice A_t. The decision model takes two inputs to make the decision: the advice set A_t from advisors and the trustworthiness vector $\vec{\tau}_t$ from the trustworthiness model. Moreover, the decision model outputs the decision d_t and an update value vector $\vec{i}_t$. This vector is then passed to the trustworthiness model to update the trustworthiness values of the advisors for the subsequent problem $t+1$.

6.1.1 Problem Formalization

For each problem, the decision maker has to make decisions based on the advice of advisors. In the binary decision problem, we recall that the decision is made between "positive" and "negative". For the decision among multiple options, the outcome of each problem is the decision d_t from a fixed set of candidate options $O = \{1, 2, 3, ..., n\}$. Similar to the binary decision problem, each advisor in advisor set $Y_t \subseteq X$ will respond with an option. For each option $o \in O$, $A^o_{Y_t} \subseteq Y_t$ is the set of advisors that state option

o is the correct one. For any two $A^i_{Y_t}$, $A^j_{Y_t}$, we have $A^i_{Y_t} \cap A^j_{Y_t} = \emptyset$ for $i \neq j$ and $A^1_{Y_t} \cup A^2_{Y_t} \cup ... \cup A^n_{Y_t} = Y_t$. In addition, we recall that $\vec{\tau}_t$ is the vector containing all the advisors' trustworthiness values.

Moreover, let $A_t = \{A^1_{Y_t}, A^2_{Y_t}, ..., A^n_{Y_t}\}$ denote the advice set of problem t. We use $d_t = f(A_t, \vec{\tau}_t) \in O$ to refer to the decision-making function. If $d_t = d^*_t$, we consider that the decision is correct. Let ν denote the number of correct problems, i.e., $\nu = |\{t \in T | d^*_t = f(A_t, \vec{\tau}_t)\}|$. Our goal is to maximize the number ν by the decision-making function $f(\cdot)$.

6.1.2 Cautious Trustworthiness Model

As in Chapter 3, we utilize Subjective Logic and a cautious trustworthiness update strategy to construct our cautious trustworthiness (CT) model. Different from the BCT model (see Section 3.1.2)), the update strategy of the CT method considers balancing the conflicts of the advice of multiple options, not just binary options.

In more detail, for each advisor who selects option o, a cautious update value $i^o_t \in [0, 1]$ is used to update the evidence α^x_t and β^x_t. This value represents the expected difference in decision probability P^o_t between o and other options, indicating how confident MASDM is that option o is a better or worse choice than other options. We recall that BCT directly uses the difference between P^o_t and other options as new evidence (See Equation 3.3), while when the number of options is more than two, we need to consider the belief probability of all options. i^x_t can be expressed as:

$$i^o_t = P^o_t - \sum_{i \in O-o} \frac{P^i_t}{\sum_{j \in O-o} P^j_t} P^i_t \tag{6.1}$$

In Equation 6.1, the part after the minus sign represents the expected probability of other options except for option o and $\frac{P^i_t}{\sum_{j \in O-o} P^j_t}$ is the weight of P^i_t. For example, if a problem t has "1, 2, 3" three options, the BWVE method calculates the probability that they are right: $P^1_t = 0.7, P^2_t = 0.2, P^3_t = 0.1$. Then, $i^1_t = 0.7 - (0.2/(0.2 + 0.1) * 0.2 + 0.1/(0.2 + 0.1) * 0.1) = 0.53$; $i^2_t = 0.2 - (0.7/(0.7 + 0.1) * 0.7 + 0.1/(0.7 + 0.1) * 0.1) = -0.42$; $i^3_t = 0.1 - (0.7/(0.7 + 0.2) * 0.7 + 0.2/(0.7 + 0.2) * 0.2) = -0.49$. Therefore, for any advisor x selecting option o, evidence α^x_t and β^x_t can be updated as:

$$\begin{aligned} \alpha^x_{t+1} &\leftarrow \alpha^x_t + i^x_t \qquad \forall i^x_t > 0, \\ \beta^x_{t+1} &\leftarrow \beta^x_t + |i^x_t| \qquad \forall i^x_t < 0 \end{aligned} \tag{6.2}$$

6.1.3 Bayesian and Weighted Voting Ensemble Decision Model

In this work, we propose the BWVE method, which extends the BBWVE method (see Section 3.1.3). Compared to BBWVE, it can make decisions among multiple options, not only binary options. In MASDM, the BWVE method acts as the decision function f. To be specific, the inputs of the BWVE method are advice set A_t and trustworthiness vector $\vec{\tau}_t$, and the output is a probability distribution over options, which allows us to receive the decision d_t and the new evidence vector for updating the advisors' trustworthiness.

As in the BBWVE method, the BWVE method combines the Bayesian aggregation method with the weighted voting method by the average uncertainty of advisors' trustworthiness (see details in Section 6.1.3.3). In more detail, when the model lacks sufficient evidence to ascertain the trustworthiness of advisors, the BWVE method prioritizes the weighted voting method. As the evidence accumulates, the BWVE method transitions its trust to the Bayesian aggregation method. Similar to the BBWVE motivation experiments (see Section 3.1.3.2), we explain this design through a BWVE motivation experiment between the Bayesian aggregation method and the weighted voting method (see Section 6.1.3.2). We detail BBWVE in the later sections.

6.1.3.1 Existing Methods

In this section, we detail the Bayesian aggregation method and the weighted voting method within problems involving more than two options.

To begin with, for the Bayesian aggregation method, for each problem t, we recall that A_t denotes the advice set. Given the advice set A_t, for any option o, let $P_t^{b,o} := P_t^{b,o}(d_t^* = o|A_t)$ denote the probability $d_t^* = o$ that aggregating by the Bayesian aggregation method will yield the optimal decision. We can express $P_t^{b,o}$ as:

$$P_t^{b,o} = \frac{P_o P(A_t|d_t^* = o)}{\sum_{i\in O} P_i P(A_t|d_t^* = i)} \tag{6.3}$$

Specifically, P_o and P_i are the base rates of candidate options. As we mentioned before, if we do not have prior information, they can be set as $1/n$. In addition, from the Bayesian aggregation method, for each problem t, $P(A_t|d_t^* = o)$ represents the probability that the aggregating advice of A_t is correct under the condition $d_t^* = o$. $P(A_t|d_t^* = o)$ can be calculated by advisors' trustworthiness as follows:

$$P(A_t|d_t^* = o) = \prod_{i\in A_{Y_t}^o} \tau_t^i \prod_{j\in Y_t - A_{Y_t}^o} \left(\frac{P_o(1-\tau_t^j)}{\sum_{k\in O\setminus o_t^j} P_k}\right) \tag{6.4}$$

Specifically, we remember that in the binary decision, we directly use $1 - \tau_t^j$ to calculate the probability (see Equation 3.8 and 3.9). It represents the probability that the option selected by advisor j is wrong and the probability that another option is correct. But when the number of options is greater than 2, we cannot directly use it to calculate the probability of an option being correct. For example, for the problem t, there is an advisor x with the trustworthiness $\tau_t^x = 0.25$. Advisor x makes a decision for a problem with four options, A, B, C, and D, and advisor x selects option A. If we directly utilize $1 - \tau_t^j$ to calculate the probabilities, we can obtain $P(x|d_t^* = A) = 0.25$, $P(x|d_t^* = B) = P(x|d_t^* = C) = P(x|d_t^* = D) = 0.75$. This result indicates that options B, C, and D are more likely to be the correct option than option A, but in fact the probabilities of the four options should be equal, i.e., $P(x|d_t^* = A) = P(x|d_t^* = B) = P(x|d_t^* = C) = P(x|d_t^* = D) = 0.25$, if options are uniformly distributed. Therefore, weights need to be considered in options B, C, and D. In Equation 6.4, for each problem t, o_t^j is the option selected by advisor j, and k is the index of options except option o_t^j. Moreover, we recall that P_o and P_k are the prior probabilities of candidate options. Therefore, $\frac{P_o}{\sum_{k \in O - o_t^j} P_k}$ is the weight of option o, which represents the proportion of option o occupies in the probability of other options correct if advisor j is wrong. If the base rates of candidate options are uniformly distributed, this weight can be set to $1/(n-1)$.

Moreover, for the weighted voting method, given the advice set A_t, for any option o, let $P_t^{w,o} := P_t^{w,o}(d_t^* = o|A_t)$ denote the probability $d_t^* = o$ that aggregates by the weighted voting method, which can be expressed as:

$$P_t^{w,o} = \frac{\sum_{i \in A_{Y_t}^o} \tau_t^i}{\sum_{j \in Y_t} \tau_t^j} \tag{6.5}$$

Table 6.1 the setting of BWVE motivation experiments

setting	value
distance coefficient dis	$0, 1, ..., 99$
the mean of advisors' real accuracy μ	$0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8$
the standard deviation of advisors' real accuracy σ	$0.2, 0.3$
the average number of advisors $\bar{y}$	$5, 10, 20$
the number of options n	$2, 3, 5, 10$
total number of advisors	50
the number of problems for each experiment	$10,000$

6.1.3.2 BWVE Motivation Experiment

Similar to the experiments in Section 3.1.3.2, we conduct a comparative experiment between the Bayesian aggregation method and the weighted voting method to explain the motivation behind the design of the BWVE method. In this work, we use the same approach of Section 3.1.3.2 to build the experimental environment. Also, we added more conditions to compare the performances of the Bayesian aggregation method and the weighted voting method.

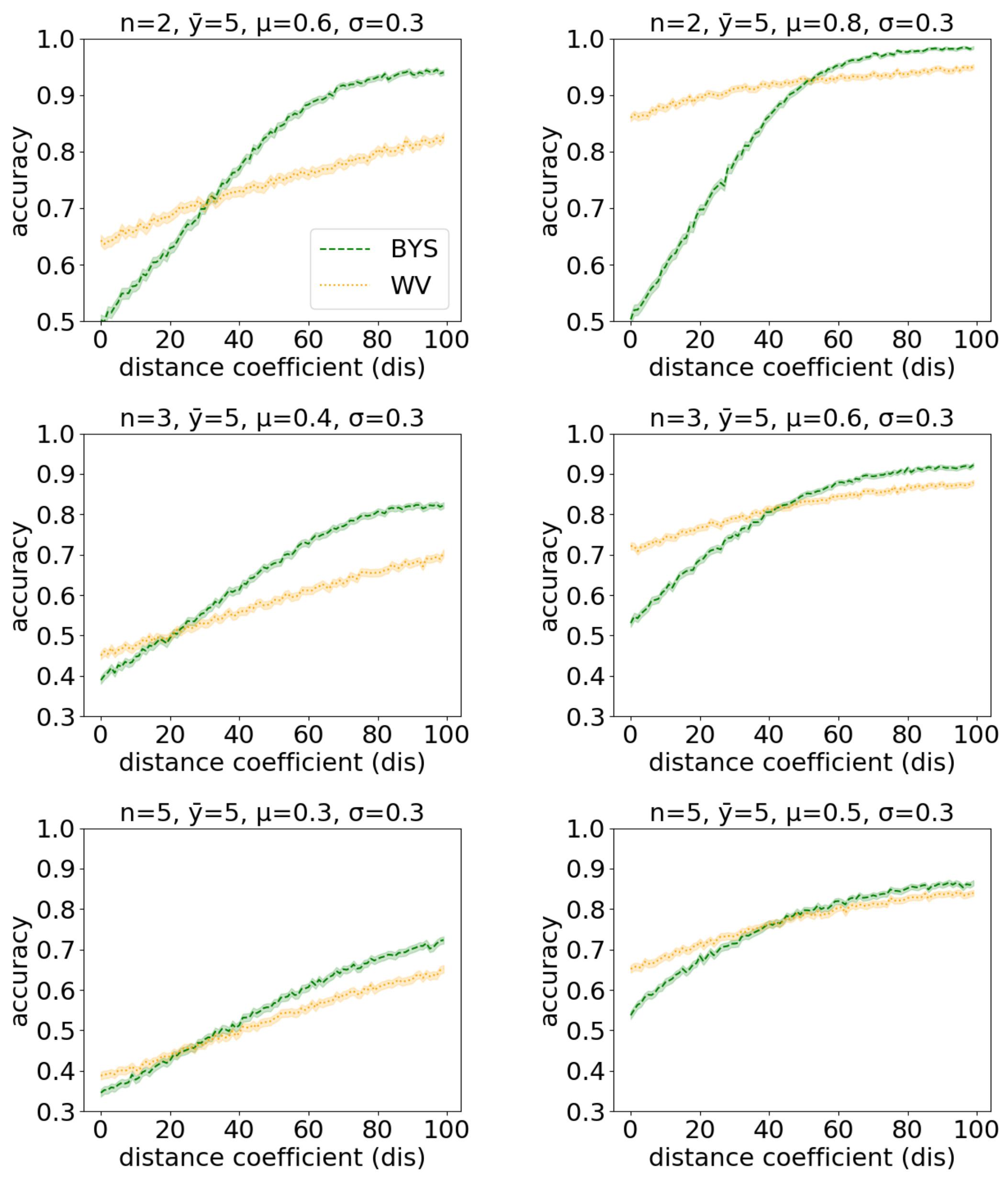


Figure 6.2 These figures show the results of the Bayesian aggregation method and the weighted voting method comparison experiments at different conditions. The X-axis of the figures depicts the distance coefficient *dis*, ranging from 0 to 100, while the Y-axis displays the average accuracy of two methods observed over $10,000$ problems. "n" indicates the number of candidate options in the problem set. "$\bar{y}$" indicates the average number of advisors selected by the model for each problem. "μ" and "σ" respectively represent the mean and standard deviation by ERGd used to generate τ_x^r.

Specifically, we use $\bar{y}$ to represent the average number of advisors per problem in each experiment. For each problem, we randomly select $\bar{y}$ advisors on average from all advisors. Furthermore, μ and σ denote the mean and standard deviation of the ERGd distribution used to generate τ_x^r. Table 6.1 provides the setting of the BWVE motivation experiment.

Figure 6.2 presents six results from the comparison experiments (more results see in Section A.1). In these results, the weighted voting method generally outperforms the Bayesian aggregation method when *dis* is small, although the advantage of the weighted voting method is modest in some results. However, as *dis* increases, the Bayesian aggregation method outperforms the weighted voting method. Therefore, we combine the Bayesian aggregation method and the weighted voting method to adapt to the varying reliability of advisors' trustworthiness and improve the overall aggregation performance.

6.1.3.3 Ensemble Decision-Making

Same as the BBWVE method (see Section 3.1.3.3), the BWVE method combines the Bayesian aggregation method and the weighted voting method by using the average uncertainty of advisors' trustworthiness as a weight to control the trust placed in each method. For each problem t, we recall that $\bar{\theta}_t$ is the average uncertainty of advisors' trustworthiness. From ensemble method, given the advice set A_t, for any option o, the probability that $d_t^* = o$ can be denote as $P_t^o := P_t^o(d_t^* = o|A_t)$, which can be expressed as:

$$P_t^o = (1 - \bar{\theta}_t)P_t^{b,o} + \bar{\theta}_t P_t^{w,o} \tag{6.6}$$

All probabilities satisfy $\sum_{o \in O} P_t^o = 1$. In addition, for each problem t, we receive the decision $d_t = \underset{o \in O}{\arg\max}\, P_t^o$.

6.2 Experiments

In this section, we evaluate MASDM through experiments in simulated environments (Section 6.2.1) and real dataset MNIST (Section 6.2.2), respectively. Similar to Section 3.2, we compare MASDM with five methods that are combined with different aggregation methods: Bayesian aggregation (BYS), Weighted Voting (WV), Bayesian and Weighted Voting Ensemble (BWVE), and trustworthiness modeling methods: Beta distribution trustworthiness model (Beta), Cautious Trustworthiness model (CT) (See Table 6.2 for the overview). Then, the combined models are (1) BYS-Beta; (2) WV-Beta; (3) BWVE-Beta; (4) BYS-CT; (5) WV-CT, and our method MASDM consists of BWVE and CT.

Table 6.2 the abbreviations of methods in MASDM experiments

abbreviations list	
BYS	Bayesian Aggregation
WV	Weighted Voting
BWVE	Bayesian and Weighted Voting Ensemble
Beta	Beta Distribution Trustworthiness Model
CT	Cautious Trustworthiness Model

6.2.1 Simulated Environment Experiments

In contrast to the MABSDM experiments where the number of advisors and options are fixed, we evaluate the performance of MASDM under different numbers of advisors and options, respectively.

6.2.1.1 Setting

Table 6.3 presents the setting of the experiments in simulated environments. In our simulated environment, we conduct experiments in four sets based on the number of candidate options n. Specifically, we considered n values of 2, 3, 5, and 10, respectively. Moreover, we generate 50 advisors with different real accuracies for each experiment. We recall that $\bar{y}$ denotes the average number of advisors for each problem. Considering the impact of the number of advisors on decision performance, for each set, we conduct three groups of experiments where $\bar{y} = 5$, 10, and 15, respectively (randomly selected from 50 advisors for each problem). In addition, we recall that τ_x^r denotes the real accuracy of advisor x. In order to observe the impact of the average real accuracy of advisors on performance, for each advisor x, τ_x^r is sampled from ERGd whose mean μ ranges in the set $\{1/n + 0.01, 1/n + 0.02, ..., 1\}$ with a fixed standard deviation $\sigma = 0.3$ (reflecting the difference of advisors' trustworthiness). The advisors need to make decisions on $1,000$ problems sequentially in each experiment with the same setting.

Table 6.3 In the experiments, we generate problem sets with four different numbers of candidate options n ($2, 3, 5$, and 10). Each problem set has three groups of experiments where the average number of advisors $\bar{y} = 5, 10, 15$. So there are a total of twelve groups of experiments. In each group of experiments, we generate the real accuracy of advisors by ERGd with the mean in the set $\{1/n + 0.01, 1/n + 0.02, ..., 1\}$.

setting	value
the number of candidate options n	$2, 3, 5, 10$
the mean of advisors' real accuracy μ	1/n+0.01, 1/n+0.02,..., 1
the average number of advisors $\bar{y}$	$5, 10, 15$
total number of advisors	50
the number of problems in each problem set	$1,000$
repeat running time of each setting	$1,000$

After making decisions on all the problems, we receive the number of correct decisions for different methods. This number is used to calculate the decision accuracy of different methods. To reduce the influence of the randomness, we run each experiment with the same settings 1,000 times (i.e., using the same mean to generate the advisors' real accuracy, the same number of candidate options, and the same number of advisors for each problem on average, and all methods use the same set of advisors in the same experiment).

Table 6.4 This table presents the mean and standard deviation of the accuracy of six methods in simulated experiments. The mean accuracy is calculated over all results (e.g., in the experiments of $n = 2$ and $\bar{y} = 5$, the mean accuracy is the mean of $50 * 1,000$ results). The standard deviation is computed from 1,000 repeated experiments.

methods		**MASDM**	**BWVE -Beta**	**BYS -CT**	**BYS -Beta**	**WV -CT**	**WV -Beta**
n = 2	**ȳ = 5**	0.8962 ±0.0144	0.8864 ±0.0174	0.8282 ±0.0318	0.807 ±0.0344	0.839 ±0.0073	0.843 ±0.0074
	ȳ = 10	0.9332 ±0.0196	0.9112 ±0.0266	0.8579 ±0.0375	0.8266 ±0.0446	0.8964 ±0.0094	0.8987 ±0.0096
	ȳ = 15	0.942 ±0.0224	0.9166 ±0.0293	0.8781 ±0.0384	0.8519 ±0.0426	0.9202 ±0.012	0.9216 ±0.0123
n = 3	**ȳ = 5**	0.8564 ±0.0088	0.8539 ±0.0093	0.823 ±0.0184	0.8526 ±0.0102	0.8075 ±0.0063	0.8133 ±0.0065
	ȳ = 10	0.9248 ±0.0122	0.9169 ±0.0151	0.8594 ±0.0283	0.9134 ±0.0162	0.8797 ±0.0076	0.8847 ±0.0079
	ȳ = 15	0.944 ±0.0138	0.9328 ±0.0172	0.8737 ±0.0293	0.9287 ±0.0186	0.9122 ±0.0085	0.9164 ±0.0089
n = 5	**ȳ = 5**	0.8164 ±0.006	0.8142 ±0.006	0.8069 ±0.0098	0.8168 ±0.0062	0.7862 ±0.0054	0.7928 ±0.0054
	ȳ = 10	0.9065 ±0.0066	0.9028 ±0.0076	0.8742 ±0.0165	0.9033 ±0.0077	0.8712 ±0.006	0.8768 ±0.0059
	ȳ = 15	0.9383 ±0.0071	0.9336 ±0.009	0.8919 ±0.021	0.9335 ±0.0093	0.909 ±0.0059	0.9141 ±0.006
n = 10	**ȳ = 5**	0.7841 ±0.0052	0.7829 ±0.0051	0.7849 ±0.0066	0.7861 ±0.0052	0.7689 ±0.0051	0.7771 ±0.0049
	ȳ = 10	0.8856 ±0.0043	0.8826 ±0.0046	0.8728 ±0.0099	0.884 ±0.0045	0.8668 ±0.0047	0.8705 ±0.0046
	ȳ = 15	0.9242 ±0.0036	0.9223 ±0.004	0.9045 ±0.0128	0.923 ±0.004	0.9078 ±0.0041	0.9103 ±0.0039

6.2.1.2 Results

Table 6.4 shows the results in each group of experiments. Overall, MASDM exhibits the best performance in almost all environments. Specifically, MASDM achieved the best performance in 10 out of the total number 12 of experiments, while only slightly lower than BYS-Beta when $n = 5, 10$ and $\bar{y} = 5$. Especially in three results where $n = 2$, MASDM exhibits an average accuracy that is approximately 8% higher than BYS-Beta.

Moreover, the CT model has the ability to further improve performance based on the BWVE method, because MASDM has higher average accuracy than BWVE-Beta in all experiments. Furthermore, MASDM exhibits a smaller standard deviation than BWVE-Beta in almost all environments, which indicates that CT is more stable than the Beta trustworthiness model. Therefore, for BWVE methods, when the distribution of the advisors' trustworthiness is unknown, using CT can receive higher expected accuracy and stability than using the trustworthiness model built by the Beta distribution.

In addition, BWVE methods exhibit greater stability than Bayesian aggregation methods. The standard deviation of the two BWVE methods is consistently smaller than that of the two Bayesian aggregation methods across all experiments, which indicates that the BWVE method has not only superior performance but also outstanding stability.

The curves in Figure 6.3 show the average accuracy of six methods as μ increases (more figures see Section A.2). In the majority of conditions, MASDM has higher accuracy than other methods, especially in cases where μ is low.

First, compared to the BWVE method, the Bayesian aggregation method is easier to mislead. In Figure 6.3, two Bayesian aggregation methods have larger error bars than the two BWVE methods. In more detail, the Bayesian aggregation method is closely dependent on the accuracy of trustworthiness. Where μ is small, it often happens that multiple untrustworthy advisors provide incorrect options during the initialization stage of the trustworthiness model, which causes the wrong option to be selected as the decision and update the trustworthiness incorrectly. Although other methods can also be misleading, due to the nature of the Bayesian aggregation method, advisors with high trustworthiness are more trusted than other methods. Therefore, these untrustworthy advisors continue to be trusted, which can undermine the accuracy of future decision-making.

Second, the weighted voting method does not effectively leverage trustworthiness when the trustworthiness model is reliable. To be specific, in the weighted voting method, the advice of multiple untrustworthy advisors usually outweighs those of trustworthy advisors. For example, in a problem with binary options "A" and "B", we assume that the advisors' trustworthiness is absolutely reliable in this problem. If 2 bad advisors with the trustworthiness of 0.5 (randomly select an option) choose "A" and one advisor with the trustworthiness of 0.9 chooses "B", the weighted voting method considers "A" as the correct response, while the Bayesian aggregation method undoubtedly chooses "B". The BWVE method prefers to leverage the Bayesian aggregation method more than the weighted voting method when the advisors' trustworthiness is reliable. Therefore, the overall performance of the BWVE method is better than the weighted voting method.

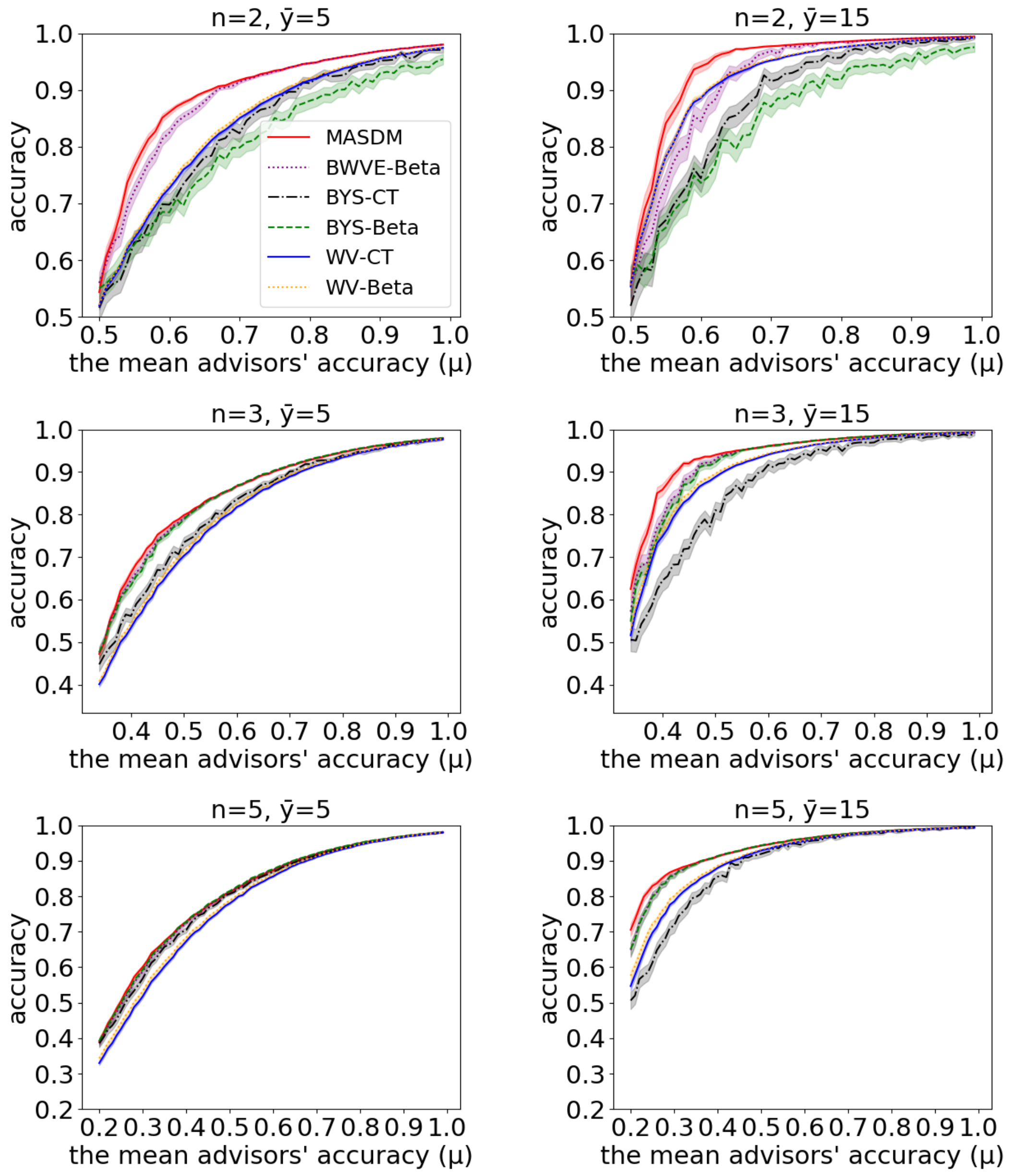


Figure 6.3 The X-axis of the figures depicts the mean of real accuracy of advisors μ, ranging from $1/n + 0.01$ (the base rate $+0.01$) to 1, while the Y-axis displays the average accuracy of six methods observed over $1,000$ experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

Third, as the number of candidate options increases, the advantage of MASDM keeps decreasing. When $\bar{y} = 5$ and $n = 5$, Bayesian aggregation methods and BWVE methods have almost the same performance. The first reason is that when $\bar{y}/n$ is relatively small, the BWVE method cannot fully exert its advantages. For example, when $\bar{y} = 5$ and $n = 5$, on average, only 1 advisor chooses each option, which means all the aggregation methods have similar performance because there is hardly aggregation. Therefore, as $\bar{y}$ increases, the advantage of MASDM becomes more significant in two results when

$n = 3, 5$ and $\bar{y} = 15$. However, there is an upper limit to the increase of this advantage, all methods have good performance when $\bar{y}$ is very large.

The second reason is that there is a saturation point in the performance of the six methods, beyond which increasing n does not significantly improve accuracy and reduces the differences between the methods. For example, when $\mu = 0.55$ and $\bar{y} = 5$, the accuracy of weighted voting methods can reach 85% when $n = 10$, while the accuracy rate is only 65% when $n = 2$. Furthermore, the increase in decision accuracy leads to the effectiveness of CT decreasing. Specifically, the design of CT is mainly to increase the stability of trustworthiness modeling. When trustworthiness is unreliable, CT can reduce wrong updates. However, the decision accuracy is high when μ is big, and it is more effective to set the step size of each updated evidence to 1 because the decision has a high probability of being correct. However, when the distribution of advisors' trustworthiness is unknown, it is more beneficial to utilize the CT model, which is also supported by the results in Table 6.4.

The third reason is that as n increases, the aggregation methods become less susceptible to being misled. In more detail, in the experimental setting, we recall that if an advisor makes a mistake, he or she randomly chooses one of the wrong options as the advice. In the problems with binary options, if the advisor is wrong, he or she can only choose the wrong option, which increases the probability that the only wrong option becomes the decision. However, when n is relatively large, such as $n = 10$, the wrong options are distributed among all wrong options, making it more difficult for the model to be misled. However, in real problems, the probability of wrong options being selected often is not evenly distributed. When the wrong advice tends to focus on a certain wrong option, the model also has the risk of being misled. In contrast, our MASDM has more advantages in such conditions because it has a higher level of robustness in preventing being misled.

We conducted paired hypotheses tests using the Mann–Whitney U test combined with Holm Bonferroni Correction to compare the results of MASDM and the five other methods. The p-values revealed that out of a total of 4,290 comparisons ($(50 + 66 + 80 + 90) \times 3 \times 5 = 4,290$), $2,170$ showed significant differences, which supports our findings. As shown in Figure 6.3, these significant differences tend to occur when the mean advisors' accuracy μ is relatively low. In addition, for cases where the mean advisors' accuracy is high, some results often do not exhibit significant differences among the methods. As we mentioned before, this is because the accuracy of the different methods approaches the saturation point, resulting in smaller gaps between them, particularly when n is large.

We also evaluate the performance of different methods in modeling trustworthiness. Figure 6.4 shows the trustworthiness error curves as the number of problems increases when μ is low (from $1/n + 0.01$ to $1/n + 0.21$). As mentioned above, when n is large

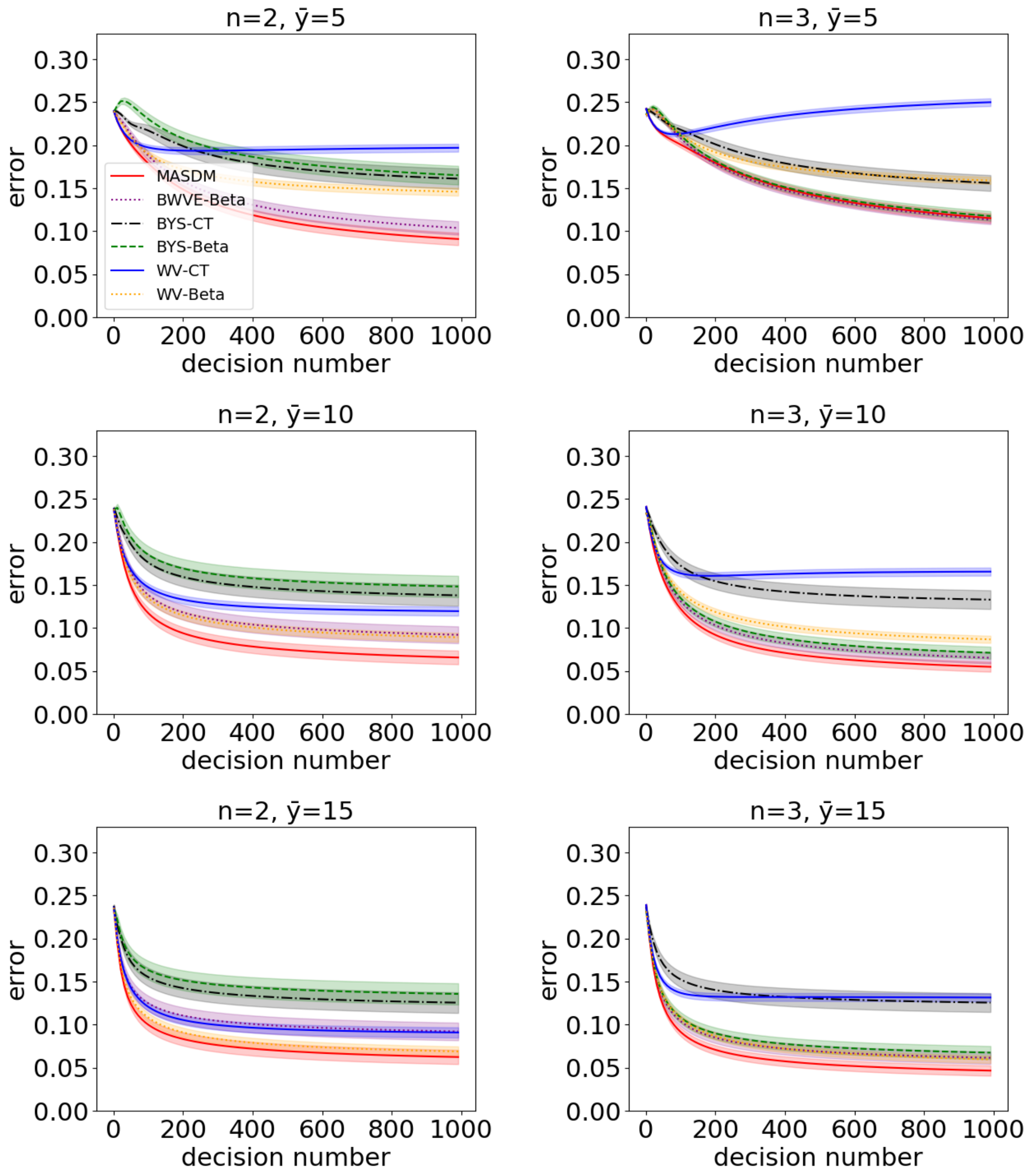


Figure 6.4 These figures show the error of advisors' trustworthiness curves from the range of low-accuracy data (from (1/n+0.01) to (1/n+0.21)). The X-axis of the figures depicts the number of problems, ranging from 1 to 1,000, while the Y-axis displays the error of the advisors' trustworthiness of six methods observed over 1,000 experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

and μ is high, the differences between most methods become small and difficult to distinguish, which is why we only show six figures in the low accuracy range in the text (all range figures and others see A.3).

Overall, our MASDM method outperforms the other methods in most conditions in terms of trustworthiness modeling. Specifically, when $n = 2$, our BWVE methods have a significant advantage over the Bayesian aggregation methods and weighted voting methods. However, when $n = 3$ and $\bar{y} = 5$, MASDM has a similar performance as BYS-Beta and BWVE-Beta. We recall that when $\bar{y}/n$ is relatively small, our BWVE method and CT model find it difficult to exert their advantage. However, this disadvantage disappears as $\bar{y}$ gradually increases, thereby we can see that MASDM outperforms the BYS-Beta method when $\bar{y} = 15$.

6.2.2 MNIST Database Experiments

In this section, we use the MNIST handwritten digit database to conduct experiments in order to test the performance of our MASDM on a real data set. MNIST is a commonly used machine learning dataset, which contains $70,000$ $28 * 28$ pixel handwritten digit pictures. These pictures come from numbers written by different people, covering digits from 0 to 9.

6.2.2.1 Setting

We randomly selected n candidate options from 10 numbers for each set of experiments, where n took on the values of 2, 3, 5, and 10, respectively. Also, for each set of experiments, we set $\bar{y} = 5$, 10, and 15. Table 6.5 presents the setting of MNIST database experiments. In addition, in order to reduce the correlation between classifiers to improve the performance of ensemble methods (Zenobi and Cunningham, 2001; Krawczyk et al., 2017), we employ seven machine learning algorithms to train 50 classifiers as the advisors. These classifiers need to sequentially classify $1,000$ randomly selected pictures. The Scikit-Learn library (Pedregosa et al., 2011) is used to implement seven machine learning methods, which are *K-Nearest Neighbors*, *Decision Tree*, *Linear Discriminant Analysis*, *Support Vector Machine*, *Gaussian Naive Bayes*, *Multinomial Naive Bayes*, and *Logistic Regression*.

In addition, as we mentioned before, advisors (classifiers) often have different reliability because of the knowledge level or data limitation. Moreover, our method also considers making reliable decisions when there are malicious advisors. Therefore, to evaluate the ability to model the trustworthiness and recognize malicious advisors, we classify advisors into three groups: good advisors, lazy advisors, and bad advisors. Specifically, good advisors are trained on $50 * n$ randomly selected pictures from the database without replacement and their output is their advice. For example, in a five-category problem ($n = 5$), each classifier is trained on 250 pictures. Lazy advisors randomly select one of the candidate options as their advice, while bad advisors are also trained on $50 * n$ pictures from the database without replacement but they randomly select an option as their advice except for their output. According to the different average performances of the classifiers, we conduct two sets of experiments: the low-accuracy group and the high-accuracy group. The former contains 21 good advisors trained by each algorithm three times; 15 lazy advisors; and 14 bad advisors trained by each algorithm two times. The latter contains 28 good advisors trained by each algorithm three times; 8 lazy advisors; and 14 bad advisors trained by each algorithm twice.

Table 6.5 In the experiments, we generate four problem sets that the numbers of candidate options (categories) n are $2, 3, 5$, and 10, respectively, and each problem set has three groups of experiments that $\bar{y} = 5, 10, 15$. So there are a total of twelve groups of experiments.

setting	value
the number of candidate options n	$2, 3, 5, 10$
the average number of advisors for each problem $\bar{y}$	$5, 10, 15$
total number of advisors	50
the problem number in each problem set	$1,000$
repeat running time of each setting	$1,000$

After the decision-making on $1,000$ problems (classifying $1,000$ pictures), we obtain the number of the correct decisions of different methods. Similar to simulated experiments, we repeat each experiment with the same settings $1,000$ times, and all methods use the same set of classifiers (advisors) in the same experiment setting each time.

6.2.2.2 Results

Table 6.6 shows the low-accuracy experimental results, and Table 6.7 shows the results of high-accuracy experiments (figures see Section A.5). Overall, MASDM still exhibits superior performance, which supports our results in simulated environments.

In low-accuracy experiments (Table 6.6), MASDM has the best performance out of all methods. When $n = 2, 3$, MASDM has a significant advantage over other methods. However, when $n = 5$ and 10, MASDM's performance is comparable to that of BWVE-BETA and BYS-BETA, but still slightly better than the other three methods. Furthermore, in high-accuracy experiments (Table 6.7), MASDM has the best performance when $n = 2, 3$ except when $n = 3$ and $\bar{y} = 5$, where BYS-BETA has a slight advantage. Moreover, BYS-BETA is slightly better than MASDM when $n = 5, 10$, but they almost have the same performance that the gaps between them are less than 0.0006. Overall, MASDM generally has the best performance across all experiment settings in MNIST database experiments, although its advantage is slightly narrowed for larger values of n.

In addition, similar to the results in simulated experiments, MASDM outperforms the BWVE-Beta method in almost all experiments, except for one case with $n = 3$ and $\bar{y} = 5$ in the high-accuracy experiments. This further demonstrates that the CT model can further improve the decision accuracy based on the BWVE method. Additionally, MASDM has a smaller standard deviation than BWVE-Beta in the experiments with

Table 6.6 This table presents the mean and standard deviation of results in low-accuracy experiments of the MNIST database. The mean and standard deviation are calculated by the results of 1,000 repeated experiments.

methods		**MASDM**	**BWVE -Beta**	**BYS -CT**	**BYS -Beta**	**WV -CT**	**WV -Beta**
$\mathbf{n = 2}$	$\bar{\mathbf{y}} = \mathbf{5}$	0.8905 ±0.1453	0.7829 ±0.3097	0.6754 ±0.395	0.6323 ±0.4082	0.7322 ±0.0431	0.7458 ±0.0522
	$\bar{\mathbf{y}} = \mathbf{10}$	0.9183 ±0.1776	0.7574 ±0.3779	0.7084 ±0.4085	0.6481 ±0.4343	0.8806 ±0.0354	0.8793 ±0.0602
	$\bar{\mathbf{y}} = \mathbf{15}$	0.9223 ±0.1939	0.7577 ±0.3886	0.7279 ±0.4073	0.6819 ±0.43	0.9215 ±0.0265	0.9007 ±0.128
$\mathbf{n = 3}$	$\bar{\mathbf{y}} = \mathbf{5}$	0.8489 ±0.029	0.847 ±0.042	0.7091 ±0.3061	0.8444 ±0.0736	0.7633 ±0.0351	0.777 ±0.0352
	$\bar{\mathbf{y}} = \mathbf{10}$	0.9103 ±0.0229	0.9031 ±0.0855	0.7482 ±0.3439	0.8904 ±0.1365	0.8732 ±0.0274	0.8776 ±0.027
	$\bar{\mathbf{y}} = \mathbf{15}$	0.9293 ±0.0206	0.919 ±0.1005	0.7974 ±0.322	0.9092 ±0.1376	0.9066 ±0.0235	0.9083 ±0.0233
$\mathbf{n = 5}$	$\bar{\mathbf{y}} = \mathbf{5}$	0.77 ±0.027	0.7675 ±0.0275	0.7193 ±0.1763	0.7695 ±0.0276	0.7362 ±0.0295	0.7426 ±0.0285
	$\bar{\mathbf{y}} = \mathbf{10}$	0.8529 ±0.0232	0.8527 ±0.0234	0.7654 ±0.2553	0.8531 ±0.0233	0.8367 ±0.0243	0.8383 ±0.024
	$\bar{\mathbf{y}} = \mathbf{15}$	0.881 ±0.0212	0.881 ±0.0212	0.8026 ±0.2485	0.8812 ±0.0211	0.8697 ±0.0225	0.8706 ±0.0222
$\mathbf{n = 10}$	$\bar{\mathbf{y}} = \mathbf{5}$	0.6687 ±0.0168	0.6659 ±0.0171	0.6452 ±0.0995	0.6667 ±0.0168	0.6556 ±0.0176	0.6614 ±0.0169
	$\bar{\mathbf{y}} = \mathbf{10}$	0.773 ±0.0134	0.7725 ±0.0135	0.74 ±0.1484	0.7728 ±0.0136	0.7664 ±0.0135	0.7684 ±0.0134
	$\bar{\mathbf{y}} = \mathbf{15}$	0.8116 ±0.0123	0.8114 ±0.0125	0.7709 ±0.1742	0.8115 ±0.0125	0.8068 ±0.0123	0.808 ±0.0125

$n = 2$ and 3, while the standard deviations are similar for both methods when $n = 5$ and 10. Therefore, in general, the CT model is more stable than the Beta trustworthiness model for the BWVE method. Furthermore, compared to Bayesian aggregation methods, MASDM is more robust for protecting against misleading. Specifically, in experiments where $n = 2$, the average accuracy of MASDM is approximately 25% higher than BYS-Beta.

However, in MNIST database experiments, the advantages of MASDM over other methods are not as pronounced as in simulation experiments when $n > 2$ in high-accuracy experiments. We believe that there are two main reasons for this. First, the correlation between advisors (classifiers) reduces the performance of aggregation methods, especially those that incorporate Bayesian aggregation methods. Although we attempted to reduce the impact of correlation by using multiple algorithms and training the advisors

Table 6.7 This table presents the mean and standard deviation of results in high-accuracy experiments of the MNIST database. More information see the caption of Table 6.6

methods		MASDM	BWVE -Beta	BYS -CT	BYS -Beta	WV -CT	WV -Beta
$n = 2$	$\bar{y} = 5$	0.9451 ±0.0241	0.9109 ±0.1738	0.8112 ±0.3253	0.7463 ±0.3746	0.8603 ±0.0351	0.8676 ±0.0364
	$\bar{y} = 10$	0.965 ±0.0194	0.9203 ±0.201	0.8564 ±0.3013	0.803 ±0.355	0.9298 ±0.0247	0.931 ±0.0251
	$\bar{y} = 15$	0.971 ±0.0177	0.9233 ±0.2081	0.8766 ±0.2843	0.8363 ±0.3313	0.9481 ±0.0207	0.9485 ±0.0211
$n = 3$	$\bar{y} = 5$	0.8909 ±0.0257	0.8913 ±0.026	0.7921 ±0.2741	0.892 ±0.0376	0.8487 ±0.0292	0.8524 ±0.0289
	$\bar{y} = 10$	0.9295 ±0.0214	0.9287 ±0.0354	0.8416 ±0.2699	0.9271 ±0.0542	0.9104 ±0.0234	0.9111 ±0.0233
	$\bar{y} = 15$	0.9416 ±0.0199	0.9408 ±0.0354	0.8789 ±0.2344	0.94 ±0.0458	0.9287 ±0.0215	0.929 ±0.0215
$n = 5$	$\bar{y} = 5$	0.8251 ±0.0253	0.824 ±0.0257	0.7998 ±0.1339	0.8257 ±0.0256	0.8061 ±0.0261	0.8078 ±0.0259
	$\bar{y} = 10$	0.8812 ±0.0216	0.8812 ±0.0216	0.8525 ±0.1558	0.8815 ±0.0216	0.8714 ±0.0224	0.872 ±0.0224
	$\bar{y} = 15$	0.9006 ±0.02	0.9006 ±0.0201	0.8704 ±0.1625	0.9007 ±0.0201	0.8935 ±0.0205	0.8939 ±0.0205
$n = 10$	$\bar{y} = 5$	0.7334 ±0.0145	0.7322 ±0.0147	0.7273 ±0.0344	0.7331 ±0.0146	0.7258 ±0.0149	0.7289 ±0.0146
	$\bar{y} = 10$	0.8087 ±0.0128	0.8086 ±0.0128	0.7823 ±0.1395	0.8089 ±0.0128	0.8047 ±0.013	0.8056 ±0.013
	$\bar{y} = 15$	0.8367 ±0.0126	0.8366 ±0.0126	0.8132 ±0.1367	0.8366 ±0.0126	0.8337 ±0.0124	0.8341 ±0.0125

with different datasets, some correlations still persist between advisors. Second, different problems (i.e. classifying the pictures) have varying levels of difficulty. In the simulated experiments, the accuracy of advisors in deciding each problem is fixed, while the accuracy fluctuates due to the difficulty of the problem in the MNIST dataset. If some pictures are more difficult or easier to classify, the trustworthiness of the advisors becomes less useful, which reduces the performance of our method. Nevertheless, the results on the MNIST database are generally consistent with the results in simulation experiments, indicating that our MASDM method has the best overall performance.

Similar to simulated experiments, we conduct paired hypotheses tests on the results of MASDM and five other methods by the Mann–Whitney U test combined with the Holm Bonferroni Correction. The p-value shows that there are $42/60$ ($12 * 5 = 60$) groups of results in low-accuracy experiments and $34/60$ groups of results in high-accuracy experiments that have significant differences, which supports our findings. However, as we analyzed earlier, the performance of MASDM in the real dataset is

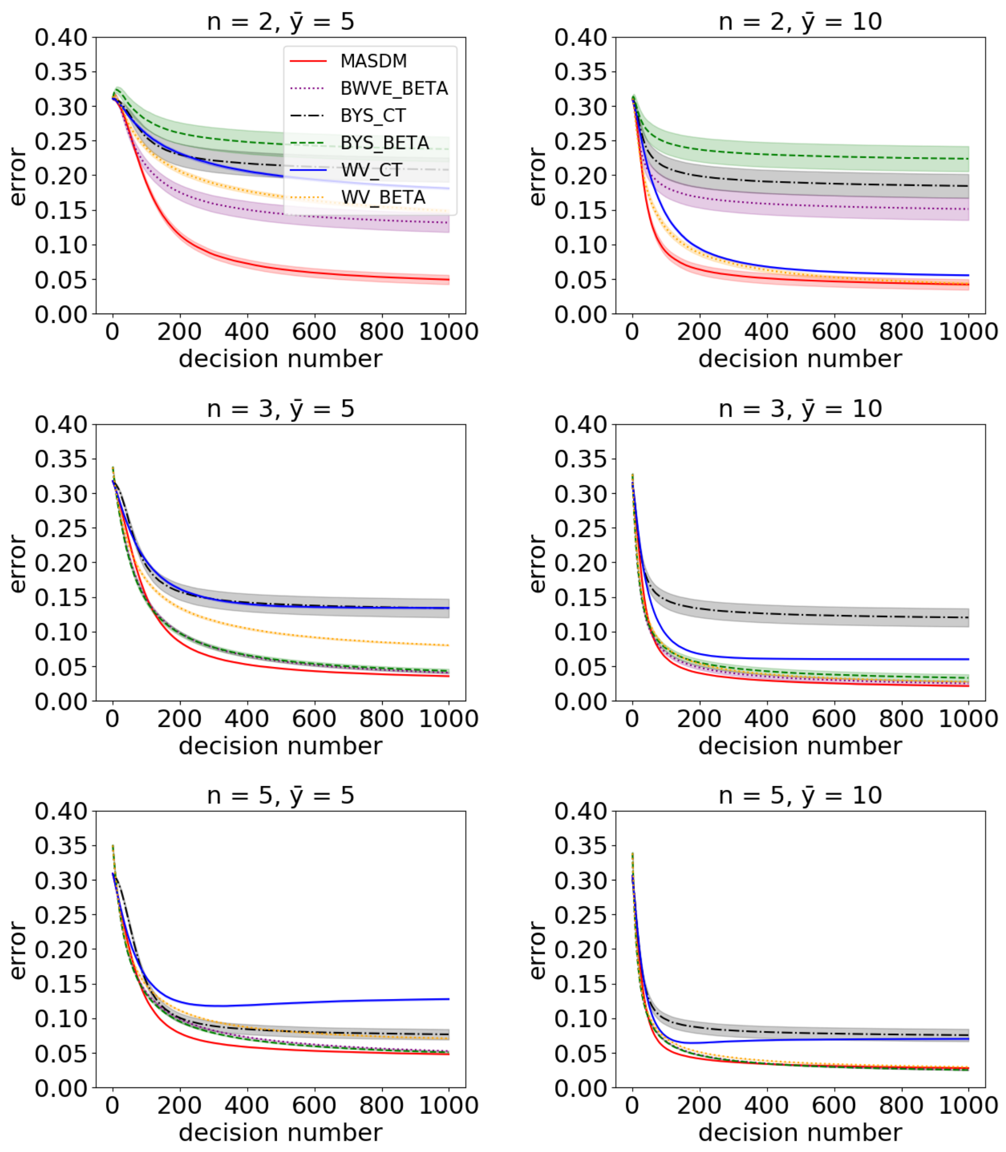


Figure 6.5 This is trustworthiness modeling results of low-accuracy experiments. The X-axis of the figures depicts the number of problems, ranging from 1 to 1,000, while the Y-axis displays the error of the advisors' trustworthiness of six methods observed over 1,000 experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

affected. In more detail, we do not find significant differences between the results of MASDM and BWVE-Beta in high-accuracy experiments. Furthermore, the results with significant differences between MASDM and BYS-Beta are concentrated in experiments where $n = 2$.

Moreover, we compare the ability of different methods to model trustworthiness. Figure 6.5 and 6.6 show 6 representative results that the average error of advisors' trustworthiness as the number of problems increases in low-accuracy and high-accuracy experiments, respectively (more figures see Section A.5). Our method outperforms other methods in most conditions in terms of the ability to model trustworthiness. Specifically, MASDM exhibits a significant advantage over other methods when $n = 2$. In addition, MASDM generally models advisors' trustworthiness faster than other methods and performs either slightly better or similar to other methods when $n > 2$.

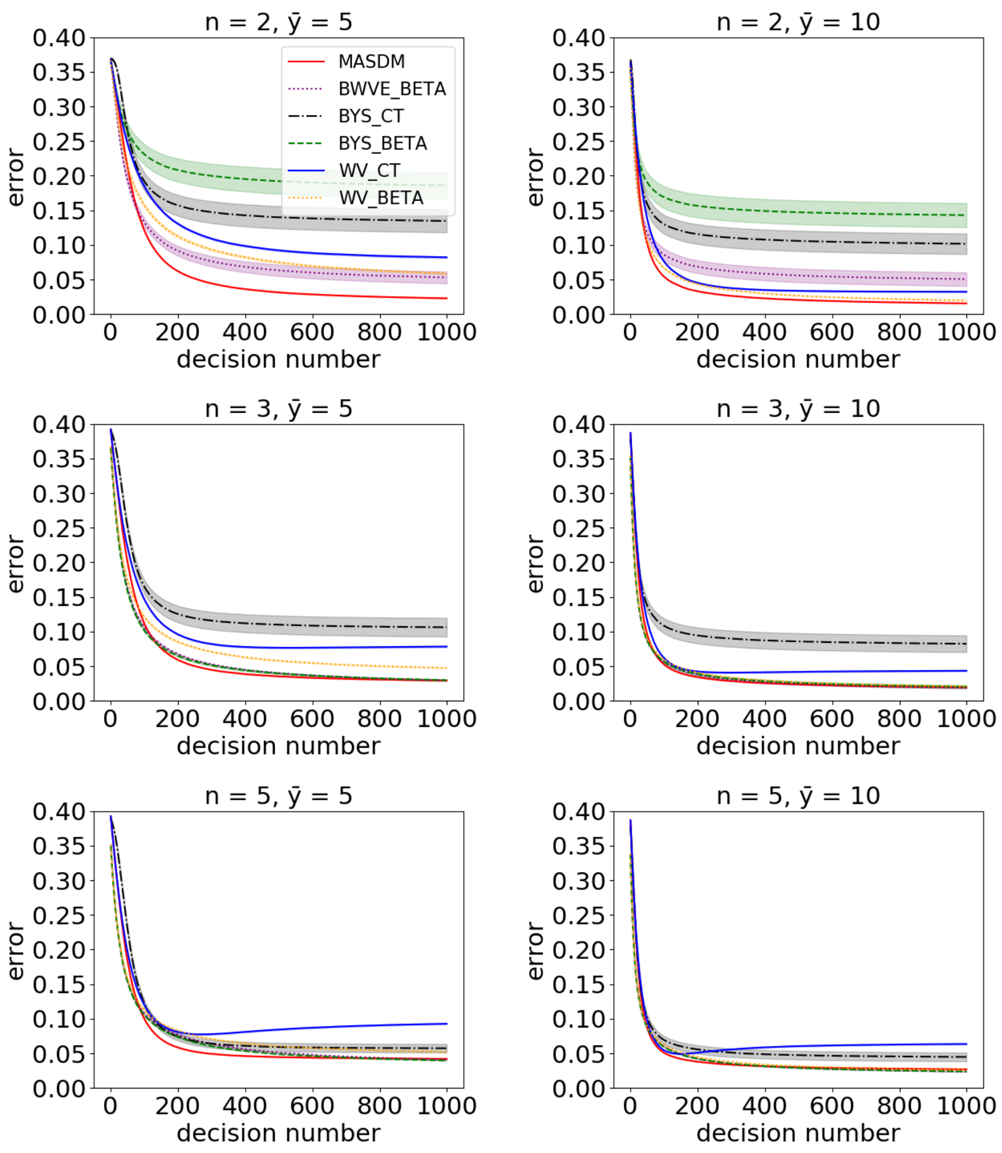


Figure 6.6 This is the trustworthiness modeling results of high-accuracy experiments. The X-axis of the figures depicts the number of problems, ranging from 1 to 1,000, while the Y-axis displays the error of the advisors' trustworthiness of six methods observed over 1,000 experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

6.3 Summary

In this chapter, we address Objective **O4.1** and **O4.2** that propose (MASDM) method, a novel approach for making decisions in sequential decision-making settings. Compared to MABSDM, it can make decisions among multiple options, not only binary options. Specifically, MASDM (1) makes decisions by aggregating the advice of multiple advisors without ground truth, (2) automatically adjusts the aggregation strategy according to the uncertainty of the advisors' trustworthiness and, (3) sequentially models the advisors' trustworthiness without prior information. Furthermore, we conduct experiments on both the simulated environments and the MNIST database to evaluate our approach. We compare our MASDM with five benchmark methods that are combined with state-of-the-art aggregating methods and trustworthiness models. Our results show that MASDM significantly outperforms five benchmark methods using state-of-the-art models.

Chapter 7

Conclusions and Future Work

This thesis is concerned with the challenge of multi-advisor sequential decision-making problems. To this end, we present several studies that address distinct aspects of this broad challenge. In addition, we provide three future research directions related to multi-advisor sequential decision-making without ground truth.

7.1 Conclusions

Following Objective **O1.1**, **O1.2** (see Section 1.3.1), we present a novel method, Multi-Advisor Binary Sequential Decision-Making (MABSDM) for binary sequential decision-making problems in Chapter 3. In more detail, MABSDM considers (1) modeling the advisors' trustworthiness sequentially without prior information and ground truth, (2) automatically adjusting the aggregation strategy according to the uncertainty of the advisors' trustworthiness and, (3) making optimal decisions by the advice of multiple advisors. Moreover, our results show that our aggregation method outperforms the state-of-the-art aggregation methods including majority voting, weighted voting, and the Bayesian aggregation method. In addition, the trustworthiness modeling results support that our MABSDM outperforms the BBWVE-Beta method (use the Beta distribution trustworthiness model instead of our BCT model).

Moreover, according to Objective **O2.1** and **O2.2**, we propose a novel interactive reinforcement learning system called Multi-Advisor Interactive Reinforcement Learning (MAIRL) (see Chapter 4). It consists of MABSDM and a review model. Specifically, MABSDM can aggregate the binary advice of multiple imperfect advisors into a reliable reward for agent training in a reward-sparse environment. In addition, the review model in MAIRL can correct the unreliable reward from advisors. Moreover, we conduct grid-world experiments to evaluate the binary feedback and MAIRL. The results of feedback forms experiments indicate that the binary feedback outperforms the ranking feedback, scaling feedback, and state value feedback in terms of performance and noise robustness. Furthermore, the results of the MAIRL experiments show that the policy trained by the MAIRL with the review model is closer to the optimal policy than that without a review model.

Thirdly, to address Objective **O3.1** and **O3.2**, we adopt the MABSDM method for a utility-maximizing problem in binary decision-making scenarios in Chapter 5. Specifically, based on the multi-advisor binary sequential decision-making problem, we need to balance the value of each problem and the cost of query advisors to make sequential optimal decisions. To address this objective, we propose a novel strategy, Multi-advisor dynamic decision-making (MADDM), for optimally selecting a set of advisers in a sequential binary decision-making setting, where multiple decisions need to be made over time. Specifically, our approach considers how to simultaneously (1) select advisors by balancing the advisors' costs and the value of problems, (2) learn the trustworthiness of advisers dynamically without prior information, and (3) make optimal decisions without access to the ground truth, improving this over time. Moreover, we test our method through problem-advice experiments in a simulated environment. We also introduce two benchmark methods, one using a fixed number of advisors (FNA) and another one using a fixed budget (BC), which are combined with state-of-the-art sampling and aggregating methods. The results show that MADDM significantly outperforms benchmark methods.

Finally, we turn our attention to Objective **O4.1** and **O4.2**, and we extend the MABSDM method to a general method, called Multi-Advisor Sequential Decision-Making (MASDM) (see Chapter 6). Compared to MABSDM, MASDM can handle decisions among multiple options, not just binary options. MASDM consists of two parts: a decision model and a trustworthiness model. The decision model is to make decisions using the advice of multiple advisors by Bayesian Weighted Voting Ensemble (BWVE) method. The trustworthiness model builds and updates advisors' trustworthiness using a Cautious Trustworthiness (CT) strategy. In addition, we evaluate and explain our methods through extensive experiments in simulated environments. Moreover, we apply our method to ensemble machine learning using the experiments by the MNIST database. The results show that MASDM has better decision accuracy and the ability to assess trustworthiness than the five benchmarks using state-of-the-art methods, notably achieving a maximum improvement of 22% in accuracy compared to the Bayesian aggregation methods.

7.2 Future Work

There are multiple interesting directions for future work. The first direction involves addressing problems where the candidate options have a biased distribution. For example, for a binary decision-making problem, if the positive advice accounts for 95% and the negative advice only 5%, we often think of this as a biased decision. In this example, if the advisor always selects positive advice, the trustworthiness (decision accuracy) of he or she is 95%. However, in some practice scenarios, if a problem results in a positive decision but a negative ground truth (false positive), the decision-maker needs to pay huge costs. For example, the number of cancer cases in a hospital is usually much smaller than the number of non-cancer cases. If the hospital diagnoses a cancer patient as non-cancer, the patient is likely to lose the opportunity for treatment and lose their life, and the hospital also needs to pay huge compensation. In this case, we cannot use decision accuracy to model the trustworthiness of advisors and consider more appropriate evaluation indicators based on the nature of the problem. Therefore, it is crucial to devise approaches that can appropriately consider the impact of the options distribution to improve the decision performance.

The second direction involves focusing on the problems with greater variance in difficulty. If the same decision strategy is applied to problems of varying difficulty, the overall accuracy of the decision-making process is compromised. For example, in a crowdsourcing problem of image recognition, some pictures may be difficult to identify, while others are easier. This difficulty prevents advisors' trustworthiness from accurately representing the impact of advisors on decisions, especially when the difficulty of the images varies greatly. Therefore the challenge is developing effective methods to accurately model and estimate the difficulty of each problem and integrate it into the multi-advisor decision-making framework.

The third direction involves reducing the impact of advisor correlation, which can be due to very similar knowledge levels or cheating. For example, two advisors often give similar advice, which can potentially bias the decision-making process and lead to less reliable outcomes. The challenge is modeling the correlation reasonably because it is influenced by multiple factors. One of these factors is the number of decisions. For instance, identical advice on 10 problems of two advisors should have a stronger correlation than that on 3 problems. Therefore, the challenge is to develop techniques that can eliminate or minimize the impact of advisor correlation, thereby improving the overall accuracy and reliability of the decisions.

Appendix A

MASDM Experiment Results

A.1 BWVE Motivation Experiment

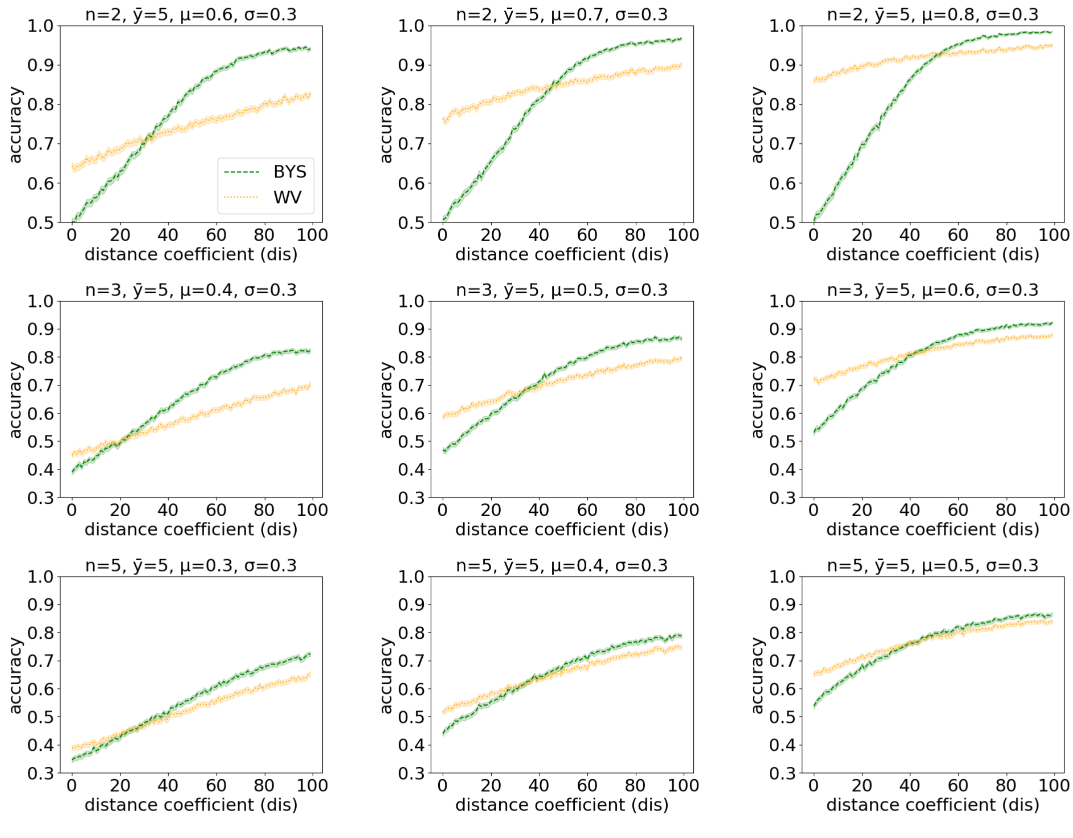


Figure A.1 These figures show the results of the Bayesian aggregation method and the weighted voting method comparison experiments at different conditions. The X-axis of the figures depicts the distance coefficient *dis*, ranging from 0 to 100, while the Y-axis displays the average accuracy of two methods observed over $10,000$ questions. "n" indicates the number of candidate options in the question set. "$\bar{y}$" indicates the average number of advisors selected by the model for each question. "μ" and "σ" respectively represent the mean and standard deviation by ERGd used to generate τ_x^r.

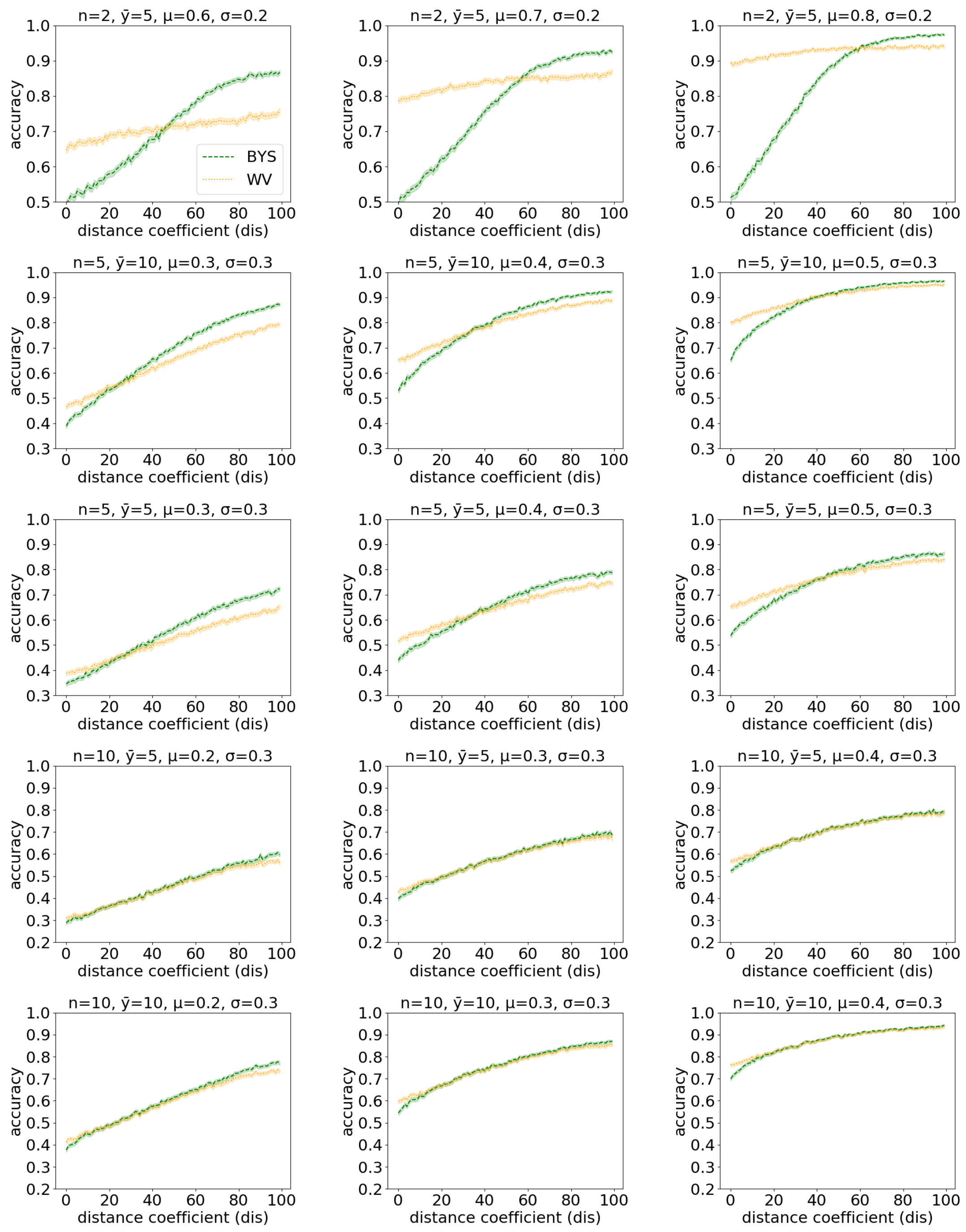


Figure A.2 These figures show the results of the Bayesian aggregation method and the weighted voting method comparison experiments at different conditions. The X-axis of the figures depicts the distance coefficient *dis*, ranging from 0 to 100, while the Y-axis displays the average accuracy of two methods observed over 10,000 questions. "n" indicates the number of candidate options in the question set. "$\bar{y}$" indicates the average number of advisors selected by the model for each question. "μ" and "σ" respectively represent the mean and standard deviation by ERGd used to generate τ_x^r.

A.2 Figures of the Accuracy Results in MASDM Simulated Experiments

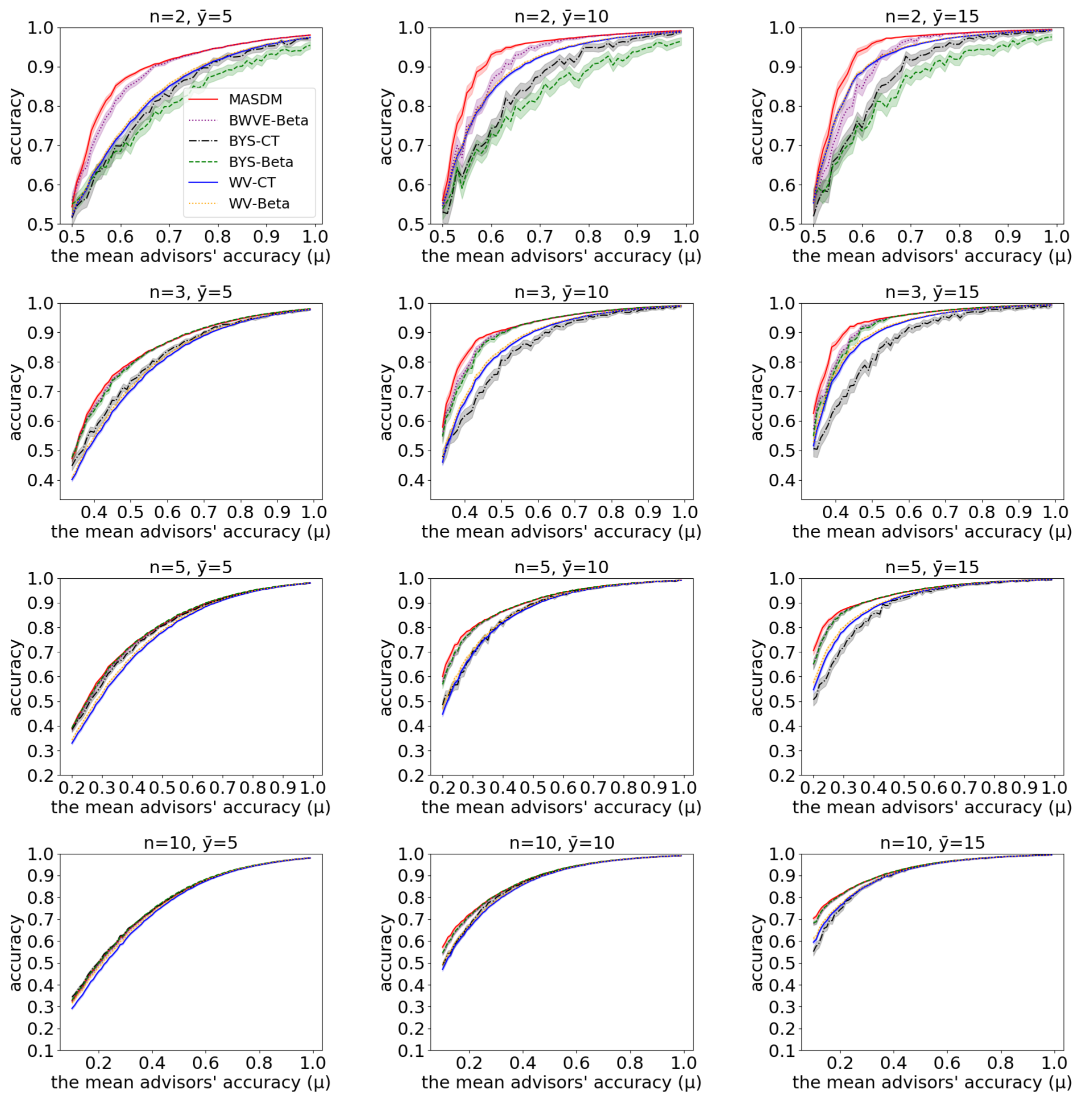


Figure A.3 The X-axis of the figures depicts the mean of real accuracy of advisors μ, ranging from $1/n + 0.01$ (the base rate $+0.01$) to 1, while the Y-axis displays the average accuracy of six methods observed over $1,000$ experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

A.3 Figures of the Trustworthiness Error Results in MASDM Simulated Experiments

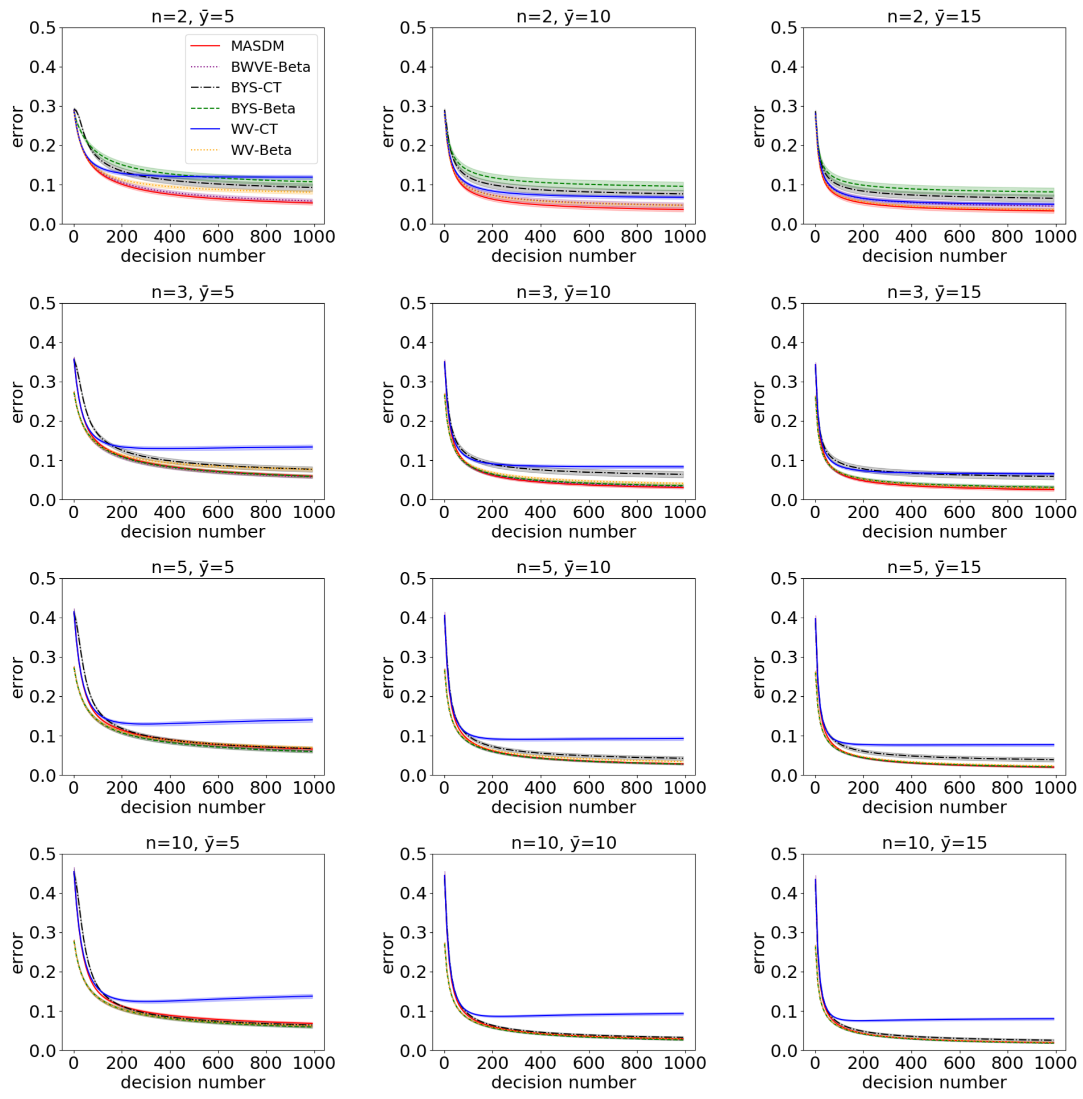


Figure A.4 These figures show the error of advisors' trustworthiness curves from all range data. The X-axis of the figures depicts the number of questions, ranging from 1 to 1,000, while the Y-axis displays the error of the advisors' trustworthiness of six methods observed over 1,000 experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

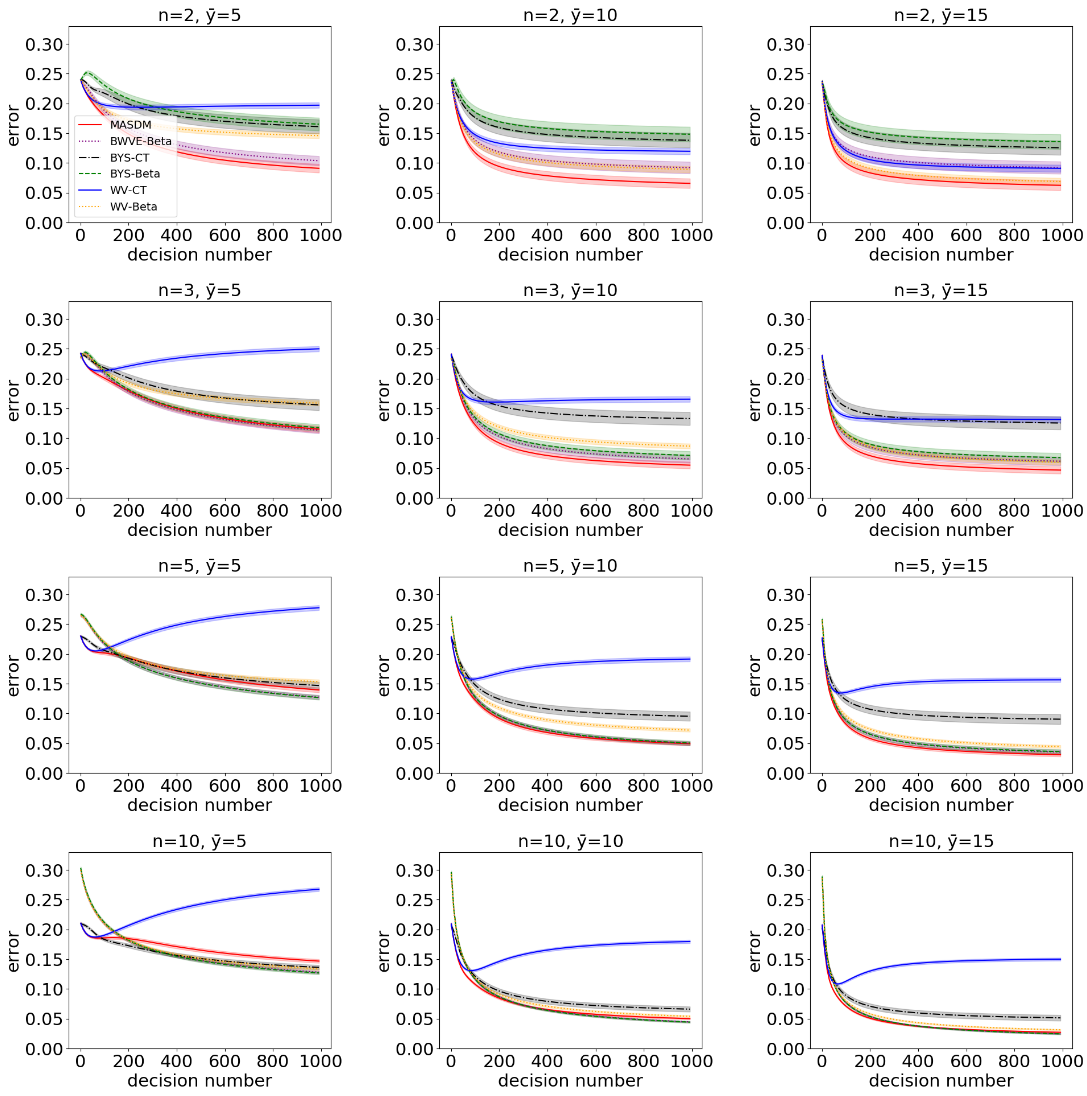


Figure A.5 These figures show the error of advisors' trustworthiness curves from the range of low-accuracy data (from (1/n+0.01) to (1/n+0.21)). The X-axis of the figures depicts the number of questions, ranging from 1 to 1,000, while the Y-axis displays the error of the advisors' trustworthiness of six methods observed over 1,000 experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar.

A.4 The Figure Results of MNIST Database Experiments

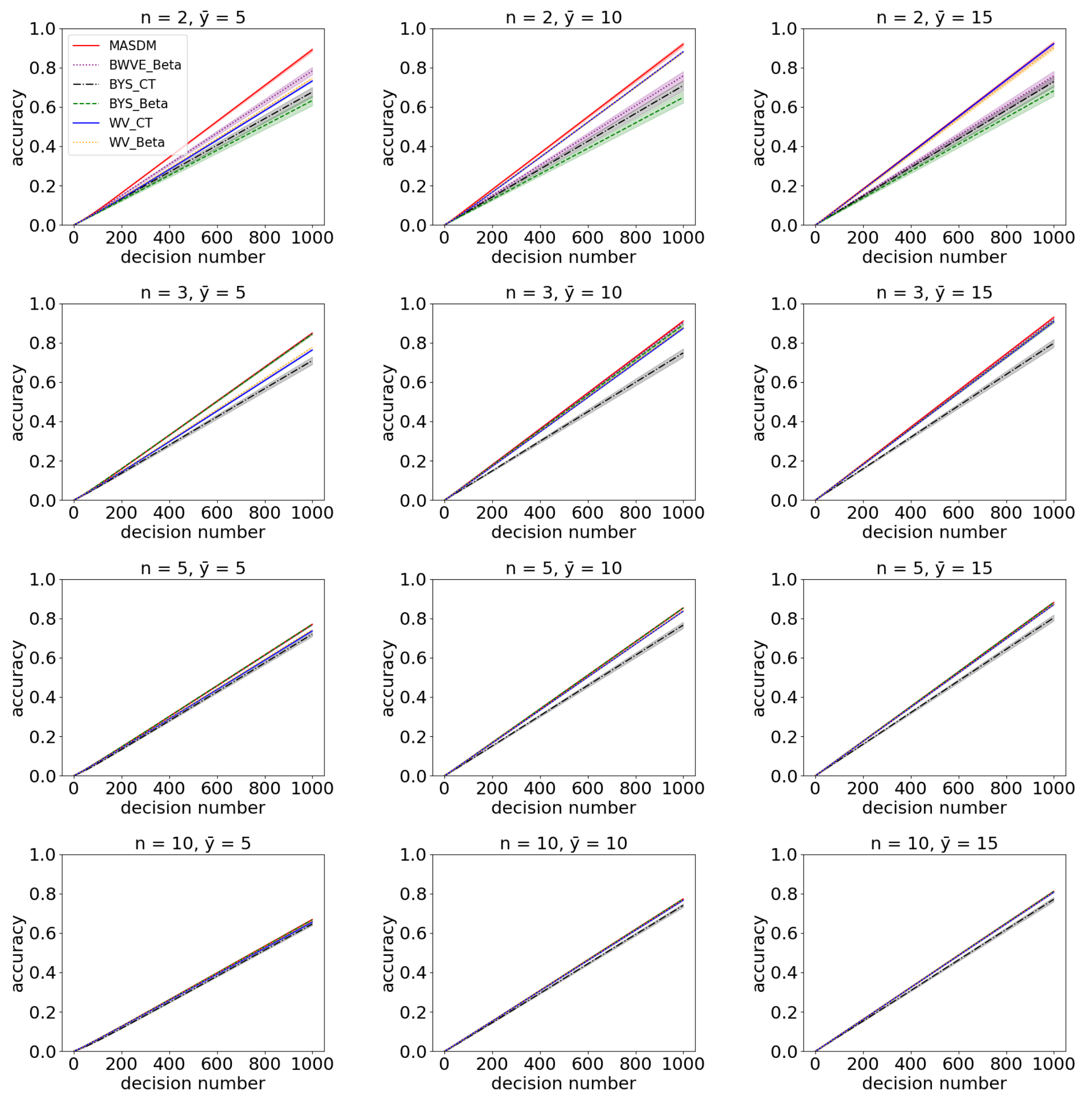


Figure A.6 The figures show the accuracy results of MNIST Database Experiments (low accuracy). The X-axis of the figures depicts the mean of real accuracy of advisors, ranging from $1/n + 0.01$ (the base rate $+0.01$) to 1, while the Y-axis displays the accuracy of six methods observed over 1,000 experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar. In the title of each figure, "n" indicates the number of candidate options in the question set, and "$\bar{y}$" indicates the average number of advisors selected by the model for each question in the group of experiments.

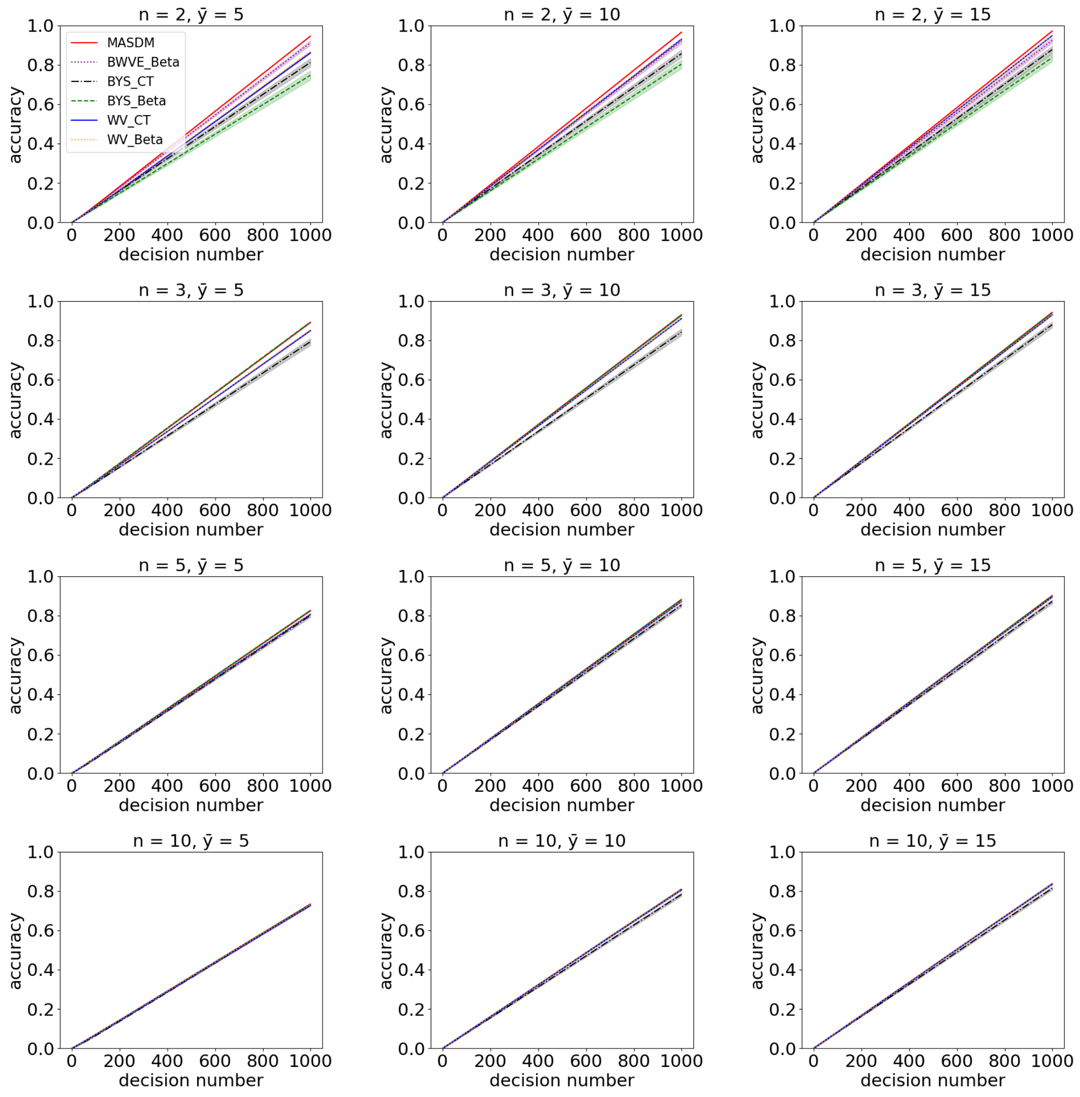


Figure A.7 The figures show the accuracy results of MNIST Database Experiments (high accuracy). More details see the caption in Figure A.6

A.5 Figures of the Trustworthiness Error Results in MNIST Database Experiments

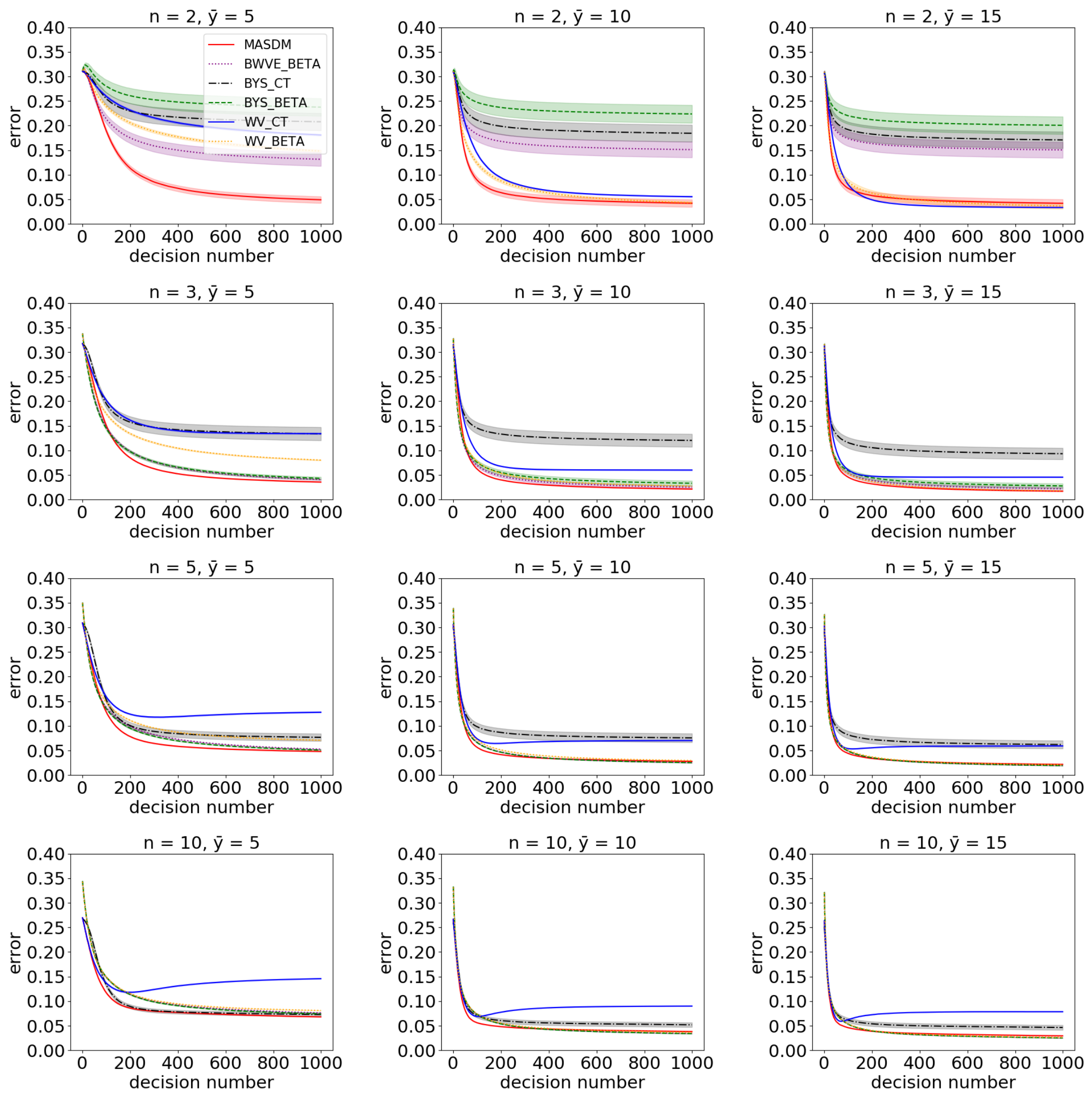


Figure A.8 The figures show the error curves of 6 methods in low-accuracy MNIST database experiments. The X-axis of the figures depicts the number of questions, ranging from 1 to 1,000, while the Y-axis displays the error of the advisors' trustworthiness of six methods observed over 1,000 experiments. The curve on the figures is accompanied by a semi-transparent region, representing the 95% confidence interval error bar. In the title of each figure, "n" indicates the number of candidate options in the question set, and "$\bar{y}$" indicates the average number of advisors for each question in the group of experiments.

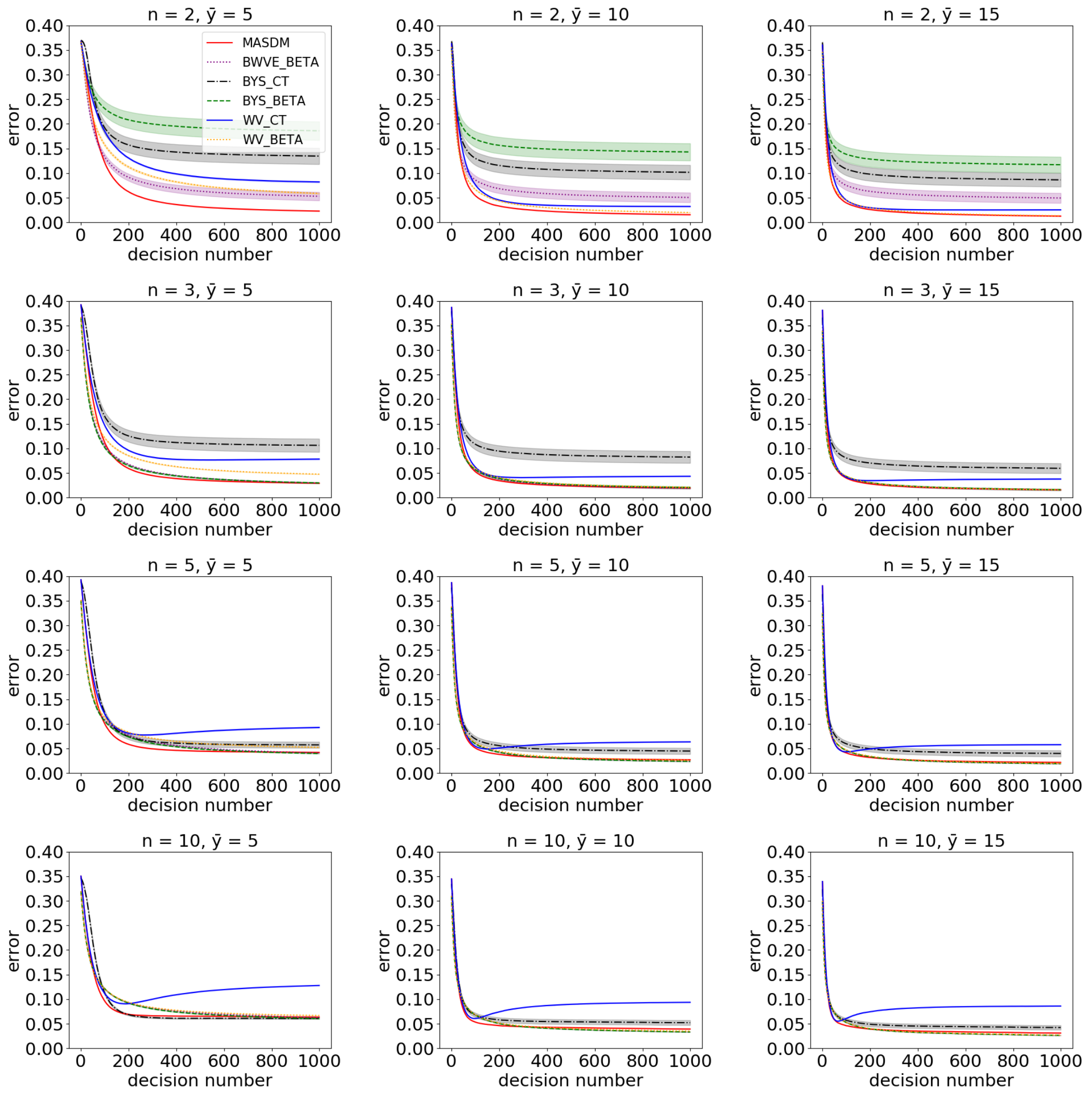


Figure A.9 The figures show the error curves of 6 methods in high-accuracy MNIST database experiments. For more information see the caption of Figure A.8

References

Ofra Amir, Ece Kamar, Andrey Kolobov, and Barbara Grosz. Interactive teaching strategies for agent training. In *In Proceedings of IJCAI 2016*, 2016.

Riku Arakawa, Sosuke Kobayashi, Yuya Unno, Yuta Tsuboi, and Shin-ichi Maeda. Dqn-tamer: Human-in-the-loop reinforcement learning with intractable feedback. *arXiv preprint arXiv:1810.11748*, 2018.

Christian Arzate Cruz and Takeo Igarashi. A survey on interactive reinforcement learning: Design principles and open challenges. In *Proceedings of the 2020 ACM designing interactive systems conference*, pages 1195–1209, 2020.

Steven E Barkan and George J Bryjak. *Fundamentals of criminal justice: A sociological view*. Jones & Bartlett Publishers, 2011.

Adam Bignold, Francisco Cruz, Richard Dazeley, Peter Vamplew, and Cameron Foale. An evaluation methodology for interactive reinforcement learning with simulated users. *Biomimetics*, 6(1):13, 2021a.

Adam Bignold, Francisco Cruz, Richard Dazeley, Peter Vamplew, and Cameron Foale. Persistent rule-based interactive reinforcement learning. *Neural Computing and Applications*, pages 1–18, 2021b.

Carlo Bonferroni. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze*, 8:3–62, 1936.

Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

Chris Burnett, Timothy J Norman, and Katia Sycara. Bootstrapping trust evaluations through stereotypes. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

Semih Cayci, Atilla Eryilmaz, and Rayadurgam Srikant. Budget-constrained bandits over general cost and reward distributions. In *International Conference on Artificial Intelligence and Statistics*, pages 4388–4398. PMLR, PMLR, 2020.

Federico Cerutti, Lance M Kaplan, Timothy J Norman, Nir Oren, and Alice Toniolo. Subjective logic operators in trust assessment: an empirical study. *Information Systems Frontiers*, 17:743–762, 2015.

Sean Chen, Jensen Gao, Siddharth Reddy, Glen Berseth, Anca D Dragan, and Sergey Levine. Asha: Assistive teleoperation via human-in-the-loop reinforcement learning. In *2022 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7505–7512. IEEE, 2022a.

Ziqi Chen, Liangxiao Jiang, and Chaoqun Li. Label augmented and weighted majority voting for crowdsourcing. *Information Sciences*, 606:397–409, 2022b.

Mingxi Cheng, Chenzhong Yin, Junyao Zhang, Shahin Nazarian, Jyotirmoy Deshmukh, and Paul Bogdan. A general trust framework for multi-agent systems. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pages 332–340, 2021.

Eugenio Chisari, Tim Welschehold, Joschka Boedecker, Wolfram Burgard, and Abhinav Valada. Correct me if i am wrong: Interactive learning for robotic manipulation. *IEEE Robotics and Automation Letters*, 7(2):3695–3702, 2022.

Melanie Coggan. Exploration and exploitation in reinforcement learning. *Research supervised by Prof. Doina Precup, CRA-W DMP Project at McGill University*, 2004.

Lizhen Cui, Jing Chen, Wei He, Hui Li, Wei Guo, and Zhiyuan Su. Achieving approximate global optimization of truth inference for crowdsourcing microtasks. *Data Science and Engineering*, 6(3):294–309, 2021a.

Yuchen Cui, Qiping Zhang, Brad Knox, Alessandro Allievi, Peter Stone, and Scott Niekum. The empathic framework for task learning from implicit human feedback. In *Conference on Robot Learning*, pages 604–626. PMLR, 2021b.

Leonardo C da Cruz, César A Sierra-Franco, Greis Francy M Silva-Calpa, and Alberto Barbosa Raposo. Enabling autonomous medical image data annotation: A human-in-the-loop reinforcement learning approach. In *2021 16th Conference on Computer Science and Intelligence Systems (FedCSIS)*, pages 271–279. IEEE, 2021.

Aida Mostafazadeh Davani, Mark Díaz, and Vinodkumar Prabhakaran. Dealing with disagreements: Looking beyond the majority vote in subjective annotations. *Transactions of the Association for Computational Linguistics*, 10:92–110, 2022.

Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28, 1979.

Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st international conference on World Wide Web*, pages 469–478, 2012.

Xibin Dong, Zhiwen Yu, Wenming Cao, Yifan Shi, and Qianli Ma. A survey on ensemble learning. *Frontiers of Computer Science*, 14:241–258, 2020.

Anestis Fachantidis, Matthew E Taylor, and Ioannis Vlahavas. Learning to teach reinforcement learning agents. *Machine Learning and Knowledge Extraction*, 1(1):21–42, 2019.

Weidong Fang, Chuanlei Zhang, Zhidong Shi, Qing Zhao, and Lianhai Shan. Btres: Beta-based trust and reputation evaluation system for wireless sensor networks. *Journal of Network and Computer Applications*, 59:88–94, 2016.

Spencer Frazier and Mark Riedl. Improving deep reinforcement learning in minecraft with action advice. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 15-1, pages 146–152, 2019.

Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer, 1996.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

Francis Galton. The ballot-box. *Nature*, 75(1952):509–510, 1907.

Meric Altug Gemalmaz and Ming Yin. Accounting for confirmation bias in crowdsourced label aggregation. In *IJCAI*, pages 1729–1735, 2021.

Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. *Advances in neural information processing systems*, 26, 2013.

Lin Guan, Mudit Verma, Sihang Guo, Ruohan Zhang, and Subbarao Kambhampati. Explanation augmented feedback in human-in-the-loop reinforcement learning. *arXiv preprint arXiv:2006.14804*, 2020.

Taha D Güneş, Timothy J Norman, and Long Tran-Thanh. Budget limited trust-aware decision making. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 101–110. Springer, 2017.

Zhaori Guo, Timothy J Norman, and Enrico H Gerding. Mtirl: Multi-trainer interactive reinforcement learning system. In *PRIMA 2022: Principles and Practice of Multi-Agent Systems: 24th International Conference, Valencia, Spain, November 16–18, 2022, Proceedings*, pages 227–242. Springer, 2022.

Zhaori Guo, Timothy J Norman, and Enrico H Gerding. Maddm: Multi-advisor decision making based on maximizing the dynamic utility. In *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023)*. International Foundation for Autonomous Agents and Multiagent Systems, 2023a.

Zhaori Guo, Timothy J Norman, and Enrico H Gerding. Multi-advisor dynamic decision-making. In *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023)*. International Foundation for Autonomous Agents and Multiagent Systems, 2023b.

Charles Isbell, Christian R Shelton, Michael Kearns, Satinder Singh, and Peter Stone. A social reinforcement learning agent. In *Proceedings of the fifth international conference on Autonomous agents*, pages 377–384, 2001.

Haein Jeon, Yewon Kim, and Bo-Yeong Kang. Interactive reinforcement learning for table balancing robot. In *Proceedings of Second International Combined Workshop on Spatial Language Understanding and Grounded Communication for Robotics*, pages 71–78, 2021.

Norman L Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. *Continuous univariate distributions, volume 2*, volume 289. John Wiley & sons, 1995.

Audun Jøsang. *Subjective logic*, volume 3. Springer, 2016.

Audun Jøsang, Jin-Hee Cho, and Feng Chen. Noninformative prior weights for dirichlet pdfs. In *2022 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 1–6. IEEE, 2022.

David Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. *Advances in neural information processing systems*, 24, 2011.

David R Karger, Sewoong Oh, and Devavrat Shah. Budget-optimal task allocation for reliable crowdsourcing systems. *Operations Research*, 62(1):1–24, 2014.

Hyun-Chul Kim and Zoubin Ghahramani. Bayesian classifier combination. In *Artificial Intelligence and Statistics*, pages 619–627. PMLR, 2012.

Josef Kittler, Mohamad Hatef, Robert PW Duin, and Jiri Matas. On combining classifiers. *IEEE transactions on pattern analysis and machine intelligence*, 20(3):226–239, 1998.

W Bradley Knox and Peter Stone. Tamer: Training an agent manually via evaluative reinforcement. In *2008 7th IEEE international conference on development and learning*, pages 292–297. IEEE, 2008a.

W Bradley Knox and Peter Stone. Tamer: Training an agent manually via evaluative reinforcement. In *2008 7th IEEE international conference on development and learning*, pages 292–297. IEEE, 2008b.

W Bradley Knox and Peter Stone. Combining manual feedback with subsequent mdp reward signals for reinforcement learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 5–12. Citeseer, 2010.

W Bradley Knox and Peter Stone. Framing reinforcement learning from human reward: Reward positivity, temporal discounting, episodicity, and performance. *Artificial Intelligence*, 225:24–50, 2015.

Bartosz Krawczyk, Leandro L Minku, João Gama, Jerzy Stefanowski, and Michał Woźniak. Ensemble learning for data stream analysis: A survey. *Information Fusion*, 37:132–156, 2017.

Samantha Krening and Karen M Feigh. Interaction algorithm effect on human experience with reinforcement learning. *ACM Transactions on Human-Robot Interaction (THRI)*, 7(2):1–22, 2018.

Samantha Krening and Karen M Feigh. Newtonian action advice: Integrating human verbal instruction with reinforcement learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 720–727, 2019.

Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2014.

Andrey Kurenkov, Ajay Mandlekar, Roberto Martin-Martin, Silvio Savarese, and Animesh Garg. AC-Teach: A bayesian actor-critic method for policy learning with an ensemble of suboptimal teachers. In *Conference on Robot Learning*, pages 717–734. PMLR, 2020.

Yann LeCun, Corinna Cortes, Chris Burges, et al. Mnist handwritten digit database, 2010.

Kimin Lee, Laura M Smith, and Pieter Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In *International Conference on Machine Learning*, pages 6152–6163. PMLR, 2021.

Guangliang Li, Hayley Hung, Shimon Whiteson, and W Bradley Knox. Using informative behavior to increase engagement in the tamer framework. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 909–916, 2013.

Guangliang Li, Hamdi Dibeklioğlu, Shimon Whiteson, and Hayley Hung. Facial feedback for reinforcement learning: a case study and offline analysis using the tamer framework. *Autonomous Agents and Multi-Agent Systems*, 34:1–29, 2020.

Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.

Quanyi Li, Zhenghao Peng, and Bolei Zhou. Efficient learning of safe driving policy via human-ai copilot optimization. *arXiv preprint arXiv:2202.10341*, 2022.

Siyuan Li and Chongjie Zhang. An optimal online method of selecting source policies for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32-1, 2018.

Yuan Li, Benjamin Rubinstein, and Trevor Cohn. Exploiting worker correlation for label aggregation in crowdsourcing. In *International conference on machine learning*, pages 3886–3895. PMLR, 2019.

Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

Jiacheng Liu, Feilong Tang, Long Chen, and Yanmin Zhu. Exploiting predicted answer in label aggregation to make better use of the crowd wisdom. *Information Sciences*, 574:66–83, 2021.

Qiang Liu, Jian Peng, and Alexander T Ihler. Variational inference for crowdsourcing. *Advances in neural information processing systems*, 25, 2012.

Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55, 2017.

James MacGlashan, Mark K Ho, Robert Loftin, Bei Peng, Guan Wang, David L Roberts, Matthew E Taylor, and Michael L Littman. Interactive learning from policy-dependent human feedback. In *International Conference on Machine Learning*, pages 2285–2294. PMLR, 2017.

Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.

Harry Markowitz. *Portfolio Selection*, volume 7-1. JSTOR, 1952.

Xiaoye Miao, Huanhuan Peng, Yunjun Gao, Zongfu Zhang, and Jianwei Yin. On dynamically pricing crowdsourcing tasks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2022.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

Andrew W Palmer, Andrew J Hill, and Steven J Scheding. Methods for stochastic collection and replenishment (scar) optimisation for persistent autonomy. *Robotics and Autonomous Systems*, 87:51–65, 2017.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 2011. URL `http://scikit-learn.org`.

Michael Peter Perrone. *Improving regression estimation: Averaging methods for variance reduction with extensions to general convex measure optimization*. Brown University, 1993.

Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

Nasim Sabetpour, Adithya Kulkarni, Sihong Xie, and Qi Li. Truth discovery in sequence labels from crowds. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 539–548. IEEE, 2021.

Murat Şensoy, Achille Fokoue, Jeff Z Pan, Timothy J Norman, Yuqing Tang, Nir Oren, and Katia Sycara. Reasoning about uncertain information and conflict resolution through trust revision. In *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

Andrew Stern, Adam Frank, and Ben Resner. Virtual petz (video session) a hybrid approach to creating autonomous, lifelike dogz and catz. In *Proceedings of the second international conference on Autonomous agents*, pages 334–335, 1998.

Halit Bener Suay and Sonia Chernova. Effect of human guidance and state space size on interactive reinforcement learning. In *2011 Ro-Man*, pages 1–6. IEEE, 2011.

Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in neural information processing systems*, pages 1038–1044, 1996.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Ana C Tenorio-Gonzalez, Eduardo F Morales, and Luis Villasenor-Pineda. Dynamic reward shaping: training a robot by voice. In *Ibero-American conference on artificial intelligence*, pages 483–492. Springer, 2010.

Andrea L Thomaz and Cynthia Breazeal. Teachable robots: Understanding human teaching behavior to build more effective robot learners. *Artificial Intelligence*, 172 (6-7):716–737, 2008.

Andrea Lockerd Thomaz, Guy Hoffman, and Cynthia Breazeal. Real-time interactive reinforcement learning for robots. In *AAAI 2005 workshop on human comprehensible machine learning*, 2005.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

Thomas L Thorpe. Vehicle traffic light control using sarsa. In *Online]. Available: citeseer. ist. psu. edu/thorpe97vehicle. html*. Citeseer, 1997.

Hamid R Tizhoosh. Reinforcement learning based on actions and opposite actions. In *International conference on artificial intelligence and machine learning*, volume 414, 2005.

Yongxin Tong, Libin Wang, Zimu Zhou, Lei Chen, Bowen Du, and Jieping Ye. Dynamic pricing in spatial crowdsourcing: A matching-based approach. In *Proceedings of the 2018 International Conference on Management of Data*, pages 773–788, 2018.

Long Tran-Thanh, Archie Chapman, Alex Rogers, and Nicholas Jennings. Knapsack based optimal policies for budget–limited multi–armed bandits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26-1, pages 1134–1140, 2012.

Long Tran-Thanh, Sebastian Stein, Alex Rogers, and Nicholas R Jennings. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. *Artificial Intelligence*, 214:89–111, 2014.

Matteo Venanzi, John Guiver, Gabriella Kazai, Pushmeet Kohli, and Milad Shokouhi. Community-based bayesian aggregation models for crowdsourcing. In *Proceedings of the 23rd international conference on World wide web*, pages 155–164, 2014.

Huiyang Wang, Diep N Nguyen, Dinh Thai Hoang, Eryk Dutkiewicz, and Qingqing Cheng. Real-time crowdsourcing incentive for radio environment maps: A dynamic pricing approach. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2018.

Garrett Warnell, Nicholas Waytowich, Vernon Lawhern, and Peter Stone. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32-1, 2018.

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.

Ming Wu, Qianmu Li, Fei Yang, Jing Zhang, Victor S Sheng, and Jun Hou. Learning from biased crowdsourced labeling with deep clustering. *Expert Systems with Applications*, 211:118608, 2023.

Yingce Xia, Haifang Li, Tao Qin, Nenghai Yu, and Tie-Yan Liu. Thompson sampling for budgeted multi-armed bandits. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

Sheng Kung Michael Yi, Mark Steyvers, Michael D Lee, and Matthew J Dry. The wisdom of the crowd in combinatorial problems. *Cognitive science*, 36(3):452–470, 2012.

Gabriele Zenobi and Padraig Cunningham. Using diversity in preparing ensembles of classifiers based on different feature subsets to minimize generalization error. In *Machine Learning: ECML 2001: 12th European Conference on Machine Learning Freiburg, Germany, September 5–7, 2001 Proceedings 12*, pages 576–587. Springer, 2001.

Yusen Zhan, Haitham Bou Ammar, and Matthew E Taylor. Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2315–2321, 2016.

Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: Is the problem solved? *Proceedings of the VLDB Endowment*, 10(5):541–552, 2017.

Xiangyu Zhong, Xuanhua Xu, and Bin Pan. A non-threshold consensus model based on the minimum cost and maximum consensus-increasing for multi-attribute large group decision-making. *Information Fusion*, 77:90–106, 2022.

Datong Zhou and Claire Tomlin. Budget-constrained multi-armed bandits with multiple plays. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32-1, 2018.

Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. CRC press, 2012.