Eliciting Thinking Hierarchy without a Prior

Yuqing Kong* Yunqi Li Yubo Zhang Zhihuan Huang Jinzhao Wu
The Center on Frontiers of Computing Studies,
Peking University
{yuqing.kong, Liyunqi, zhangyubo18, zhihuan.huang, jinzhao.wu}@pku.edu.cn

Abstract

When we use the wisdom of the crowds, we usually rank the answers according to their popularity, especially when we cannot verify the answers. However, this can be very dangerous when the majority make systematic mistakes. A fundamental question arises: can we build a hierarchy among the answers *without any prior* where the higher-ranking answers, which may not be supported by the majority, are from more sophisticated people? To address the question, we propose 1) a novel model to describe people's thinking hierarchy; 2) two algorithms to learn the thinking hierarchy without any prior; 3) a novel open-response based crowdsourcing approach based on the above theoretic framework. In addition to theoretic justifications, we conduct four empirical crowdsourcing studies and show that a) the accuracy of the top-ranking answers learned by our approach is much higher than that of plurality voting (In one question, the plurality answer is supported by 74 respondents but the correct answer is only supported by 3 respondents. Our approach ranks the correct answer the highest without any prior); b) our model has a high goodness-of-fit, especially for the questions where our top-ranking answer is correct. To the best of our knowledge, we are the first to propose a thinking hierarchy model with empirical validations in the general problem-solving scenarios; and the first to propose a practical open-response based crowdsourcing approach that beats plurality voting without any prior.

1 Introduction

The wisdom of the crowds has been proved to lead to better decision-making and problem-solving than that of an individual, especially when we do not have sufficient prior knowledge to identify individual experts [26, 2, 28]. Plurality is one of the most popular ways to aggregate the crowd's opinions. The opinions are usually ranked according to their popularity. However, it can be very dangerous when the majority are systematically biased. Here is a real-world study we perform. We have asked multiple top university students the following question.

The radius of Circle A is 1/3 the radius of Circle B. Circle A rolls around Circle B one trip back to its starting point. How many times will Circle A revolve in total?

We have collected answers "1 (11 people), 2 (8 people), 3 (134 people), 4 (16 people), 6 (27 people), 9 (21 people)". The plurality answer is "3", the ratio between the big circle's circumference and the small circle's. However, the correct answer is "4"[1] which is only supported by 16 people.

*corresponding author

[1]Interested readers are referred to `https://math.stackexchange.com/questions/1351058/circle-revolutions-rolling-around-another-circle` for explanations.

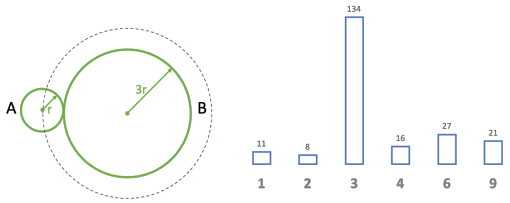


Figure 1: Collected answers of the circle problem

With prior knowledge like the expertise level of each individual respondent, we may be able to identify the correct answer. However, sometimes it's quite expensive and difficult to obtain prior knowledge, especially in new fields.

To address the above issue, Prelec et al. [19] propose an innovative approach, surprisingly popular. They prepare multiple choices, ask the respondents to pick one option, and more importantly, predict the distribution over other people's choices. They use the predictions to construct a prior distribution over the choices, and then select the choice that is more popular than the prior such that the bias is corrected. Many other work [11, 6, 20] develop the idea of using the prior or the predictions to correct the bias.

However, first, it's not applicable to employ the previous approaches into the running example, the circle problem, because they require prior knowledge to design the choices. It's also effortful for respondents to report a whole distribution over all choices.

Second, previous works focus on using the predictions to correct bias, while it's intrinsically interesting to build a thinking hierarchy using their predictions. This leads to a hierarchy among the answers as well. The famous cognitive hierarchy theory (CHT) [24, 25, 3] builds a thinking theory in the scenarios when people play games such that we can learn the actions of players of different sophistication levels. Nevertheless, CHT is designed only for a game-theoretic setting.

We are curious about building a thinking theory in general problem-solving scenarios. The key insight is that people of a more sophisticated level know the mind of lower levels, but not vice versa [3, 10]. We want to apply the insight to learn the answers of people of different sophistication levels, called the *thinking hierarchy*, without any prior.

Key question We aim to build a practical approach to learn the thinking hierarchy *without any prior*. Based on the thinking hierarchy, we can rank the answers such that the higher-ranking answers, which may not be supported by the majority, are from more sophisticated people.

In addition to building a thinking theory in the general problem-solving scenarios, in practice, there are multiple reasons why we want the hierarchy rather than only the best. First, for some questions like subjective questions (e.g. why are bar chairs high?), there may be more than one high-quality answer and the full hierarchy provides a richer result. Second, the hierarchy among the answers helps to understand how people think better, which is important especially when we try to elicit people's opinions about a policy.

Our approach We follow the framework of asking for answers and predictions simultaneously and extend it to a more practical open-response based paradigm. The paradigm asks a single open response question and asks for both each respondent's answer and prediction [2] for other people's answers. For example, in the circle problem, a respondent can provide: answer: "4", prediction: "3".

[2]Unlike previous work, the prediction in our model is not a distribution but an answer the respondent thinks other people may answer.

We then construct an answer-prediction matrix $\mathbf{A}$ that records the number of people who report a specific answer-prediction pair (e.g. Figure 2(a) shows that 28 people answer "3" and predict other people answer "6".).

To learn the thinking hierarchy, we propose a novel model. Our model describes how people of different sophistication levels answer the question and more importantly, predict other people's answers. The joint distribution over a respondent's answer and prediction depends on the latent parameters that describe people's thinking hierarchy. We show that given the joint distribution over a respondent's answer and prediction, we can infer the latent thinking hierarchy by solving a new variant of the non-negative matrix factorization problem, called non-negative congruence triangularization (NCT), which may be of an independent interest. Based on the analysis of NCT, we provide two simple answer-ranking algorithms and show that with proper assumptions, the algorithms will learn the latent thinking hierarchy given the joint distribution over a respondent's answer and prediction.

Finally, we show that the answer-prediction matrix collected by the paradigm is a proxy for the joint distribution over a respondent's answer and prediction. We implement the NCT based answer-ranking algorithms on the answer-prediction matrix. The default algorithm ranks the answers to maximize the sum of the square of the elements in the upper triangular area of the matrix. In a variant version, to allow different answers to have the same sophistication level, the answers are partitioned to compress the matrix. The algorithm maximizes the sum of the square of the upper triangular area of the compressed matrix.

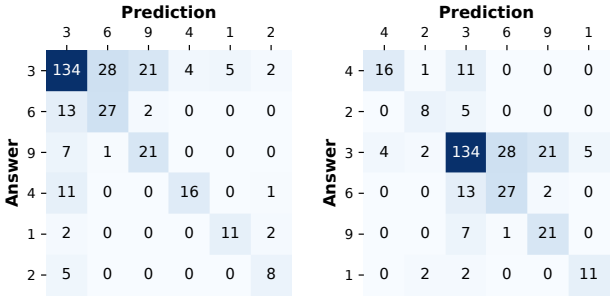
In addition to the above theoretic framework, we also run empirical studies by asking people questions in various areas including math, Go, general knowledge, and character pronunciation.

Example 1.1 (Empirical results of the circle problem)**.** *In the circle problem, we have collected the empirical answer-prediction matrix (Figure 2(a)) and ranked it (Figure 2(b)) based on the default algorithm. The default ranking algorithm does not use any diagonal element. Thus, for ease of illustration, the diagonal elements are modified (i.e., for all a, $A_{a,a}$ is modified to the number of respondents who answer a) such that we can compare our method to the plurality voting visually.*

More empirical results will be illustrated in Section 3.1. We show the superiority of our algorithm by comparing our algorithm to plurality voting by the accuracy of the top-ranking answers. We also test the goodness-of-fit of our model based on the collected data set. To summarize, we provide a novel theoretic framework to study people's thinking hierarchy in the problem-solving scenarios and a practical open-response based crowdsourcing approach that outputs a high-quality answer only supported by 16 people when the wrong answer is supported by 134 people (another question is 3 vs. 74) without any prior.

1.1 Related work

Information aggregation with the second order information Prelec [18], Prelec et al. [19] start the framework that asks the respondents both their answers and predictions for the distribution over other people's answers. However, to implement the framework, the requester needs to design a multi-choice question whose choices may require prior knowledge. The effort for a distribution report is non-minimal to many respondents and the quality can be an issue since most people are not perfect Bayesian. Kong and Schoenebeck [10] study how to elicit thinking hierarchy theoretically by people's predictions and also assumes that more sophisticated people can reason about the mind of less sophisticated people. However, they focus on multi-choice questions, and in their framework, agents either need to perform multiple tasks or report non-minimal distributions. Thus, it's difficult to perform empirical studies using their framework and there is also no empirical validation. Moreover,



(a) Original (b) Ranked

Figure 2: The empirical results of the circle problem.

they assume more sophisticated people can reason about the mind of ALL less sophisticated people while our model allows more sophisticated people cannot reason about some less sophisticated people. Two recent works Hosseini et al. [7], Schoenebeck and Tao [21] study how to aggregate people's votes to rank a set of predetermined candidates (e.g. ranking paintings based on price [7], or two presidential candidates [21]) better by using people's predictions. Both of them treat the pairwise comparisons of the candidates as the elicited signals and use people's predictions to improve the signal quality. In contrast, rather than eliciting ranking based on price or preference, we elicit the thinking hierarchy among people by using people's predictions for other people to determine the hierarchy over the answers. Like Prelec et al. [19], Hosseini et al. [7], Schoenebeck and Tao [21], there are other works that use people's predictions to reduce the bias of the collected feedback. For example, Dasgupta et al. [6] use people's predictions to reduce the bias caused by interactions between users on a social network. Rothschild and Wolfers [20] show that voters' expectations for other people's votes are more informative than their intentions. In addition to the discrete setting, recently, a growing literature, including Palley and Soll [16], Martinie et al. [12], Chen et al. [4], Wilkening et al. [29], Palley and Satopää [15], Peker [17], aggregate forecasts with additional second order information like each forecaster's expectation for the average of other forecasters' forecasts.

Peer prediction Starting from Miller et al. [13], a series of work (e.g.[13, 18, 5, 22, 11, 9]) focus on eliciting information without a prior by designing incentive-compatible mechanisms. This field is called peer prediction, or information elicitation without prior. In contrast, this work focus on how to aggregate information and identify high-quality information without a prior. Liu et al. [11], Prelec et al. [19], Hosseini et al. [7], Schoenebeck and Tao [21] focus on information aggregation without prior. However, they do not focus on the problem of learning thinking hierarchy like this work.

Bounded rationality Starting from Simon [23], the term "bounded rationality" describes the decision maker's cognitive limitations. Stahl [24] propose a behavioral model of bounded rationality to predict people's behaviors in strategic games, the level-k theory. Level-k theory assumes that players have different levels of sophistication. Level-0 players play non-strategically, level-1 players play optimal response to level-0 players... A variant of level-k theory is cognitive hierarchy theory [25, 3] where level-k players believe that lower-level players' percentages follow a certain type of distribution. Our Thinking hierarchy model is conceptually similar to level-k but focuses on a non-game setting. Moreover, the level-k theory focus on estimating the average levels of a population while we focus on identifying each respondent's level.

2 Learning Thinking Hierarchy

We first introduce our model for thinking hierarchy. Tversky and Kahneman [27] propose that people have two systems, a fast and intuitive system, and a slow and logical system. For example, Alice starts to solve the circle problem. When she reads the question, she can run her intuitive system 1 and obtain answer "3". However, when she starts to think carefully, she runs her more careful system 2 to obtain answer "4".

We propose a more general model where people can run multiple oracles to approach the question during their thinking process. Conceptually similar to the cognitive hierarchy theory [3], we assume the oracles have different levels. People usually run lower level oracles before the higher level oracles. In the previous example, system 1 is the lower level oracle and system 2 is the higher one. We assume that Alice runs system 2 after system 1. Thus, in our model, people who know the answer is "4" can predict that other people answer "3", which is hypothesized in Kong and Schoenebeck [10]. It's also possible that some people run the most sophisticated oracle directly without running the lower ones. Thus, we design the model such that it does not require the higher-type to be able to predict ALL lower types. We also allow the oracles' outputs to be random.

Section 2.1 introduces the model of thinking hierarchy. Section 2.2 reduces the model inference problem to a matrix decomposition problem. Section 2.3 and section 2.4 show how to learn the thinking hierarchy.

2.1 Thinking hierarchy

Fixing a question q (e.g. the circle problem), T denotes the set of thinking types. A denotes the set of possible answers. Both T and A are finite sets. Δ_A denotes all possible distributions over A. We sometimes use "prob" as a shorthand for probability.

Generating answers We will describe how people of different thinking types generate answers.

Definition 2.1 (Oracles of thinking types $\mathbf{W}$)**.** *An answer generating oracle maps the question to an (random) answer in A. Each type t corresponds to an oracle O_t. The output $O_t(q)$ is a random variable whose distribution is $\mathbf{w}_t \in \Delta_A$. $\mathbf{W}$ denotes a $|T| \times |A|$ matrix where each row t is $\mathbf{w}_t$.*

Each respondent is type t with prob p_t and $\sum_t p_t = 1$. A type t respondent generates the answer by running the oracle O_t. For all $a \in A$, the probability that a respondent answers a will be $\sum_t p_t \mathbf{w}_t(a)$. We assume the probability is positive for all $a \in A$.

Example 2.2 (A running example)**.** *There are two types $T = \{0, 1\}$. The answer space is $A = \{3, 4, 6\}$. O_0 will output '3' with probability 0.8 and '6' with probability 0.2. O_1 will output '4' deterministically. In this example, $\mathbf{W} = \begin{bmatrix} 0.8 & 0 & 0.2 \\ 0 & 1 & 0 \end{bmatrix}$ where the first row is the distribution of O_0's output and the second row is the distribution of O_1's output.*

Generating predictions We then describe how people of different thinking types predict what other people will answer. Here the prediction is not a distribution, but an answer other people may report. When a type t respondent makes a prediction, she will run an oracle, which is $O_{t'}$ with probability $p_{t \to t'}$ where $\sum_{t'} p_{t \to t'} = 1$. She uses the output of $O_{t'}$ as the prediction $g \in A$.

Combination: answer-prediction joint distribution M $\mathbf{M}$ denotes a $|A| \times |A|$ matrix where $M_{a,g}$ is the probability an respondent answers a and predicts g. $\mathbf{\Lambda}$ denotes a $|T| \times |T|$ matrix where $\Lambda_{t,t'} = p_t p_{t\to t'}$ is the probability a respondent is type t and predicts type t'.

Example 2.3. *In this example, when type 0 respondent makes a prediction, with prob 1, she will run O_0 again. When type 1 respondent makes a prediction, with prob 0.5, she will run O_1 again, with prob 0.5, she will run O_0. Moreover, a respondent is type 0 with prob 0.7, and type 1 with prob 0.3. Here* $\mathbf{\Lambda} = \begin{bmatrix} p_0 p_{0\to 0} & p_0 p_{0\to 1} \\ p_1 p_{1\to 0} & p_1 p_{1\to 1} \end{bmatrix} = \begin{bmatrix} 0.7 * 1 & 0.7 * 0 \\ 0.3 * 0.5 & 0.3 * 0.5 \end{bmatrix} = \begin{bmatrix} 0.7 & 0 \\ 0.15 & 0.15 \end{bmatrix}$.

Claim 2.4. *Based on the above generating processes,* $\mathbf{M} = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}$.

Proof. For each respondent, the probability she answers a and predicts g will be

$$M_{a,g} = \sum_t p_t \mathbf{w}_t(a) \sum_{t'} p_{t\to t'} \mathbf{w}_{t'}(g) = \sum_{t,t'} \mathbf{w}_t(a) p_t p_{t\to t'} \mathbf{w}_{t'}(g).$$

We sum over all possible types the respondent will be. Given she is type t, she runs oracle O_t to generate the answer and $\mathbf{w}_t(a)$ is the probability that the answer is a. We sum over all possible oracles she runs to predict. Given that she runs $O_{t'}$, $\mathbf{w}_{t'}(g)$ is the probability the prediction is g. □

Key assumption: upper-triangular Λ we assume that people of a less sophisticated level can never run the oracles of more sophisticated levels. A linear ordering of types $\pi : \{1, 2, \cdots, |T|\} \mapsto T$ maps a ranking position to a type. For example, $\pi(1) \in T$ is the top-ranking type.

Assumption 2.5. *We assume that with a proper ordering π of the types, $\mathbf{\Lambda}$ is an upper-triangular matrix. Formally, there exists π such that $\forall i > j, \mathbf{\Lambda}_{\pi(i),\pi(j)} = 0$. Any π that makes $\mathbf{\Lambda}$ upper-triangular is a valid thinking hierarchy of the types.*

In the running example, the valid thinking hierarchy is $\pi(1) =$ type 1, $\pi(2) =$ type 0. Note that the above assumption does not require $\forall i \leq j, \mathbf{\Lambda}_{\pi(i),\pi(j)} > 0$. When $\mathbf{\Lambda}$ is a diagonal matrix, types cannot predict each other and are equally sophisticated, thus any ordering is a valid thinking hierarchy.

An algorithm **finds the thinking hierarchy** when the algorithm is given $\mathbf{M}$ which is generated by latent (unknown) $\mathbf{W}, \mathbf{\Lambda}$, and the algorithm will output a matrix $\mathbf{W}^*$ which equals a row-permuted $\mathbf{W}$ where the row order is a valid thinking hierarchy. Formally, there exists a valid thinking hierarchy π such that the i^{th} row of $\mathbf{W}^*$ is the $\pi(i)^{th}$ row of $\mathbf{W}$, i.e, $\mathbf{w}^*_i = \mathbf{w}_{\pi(i)}$.

2.2 Non-negative Congruence Triangularization (NCT)

With the above model, inferring thinking hierarchy leads to a novel matrix decomposition problem, which is similar to the symmetric non-negative matrix factorization problem (NMF)[3]. A non-negative matrix is a matrix whose elements are non-negative.

Definition 2.6 (Non-negative Congruence[4] Triangularization (NCT))**.** *Given a non-negative matrix $\mathbf{M}$, NCT aims to find non-negative matrices $\mathbf{W}$ and non-negative upper-triangular matrix $\mathbf{\Lambda}$ such that $\mathbf{M} = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}$. In a Frobenius norm based approximated version, given a set of matrices $\mathcal{W}$, NCT aims to find non-negative matrices $\mathbf{W}$ and non-negative upper-triangular matrix $\mathbf{\Lambda}$ to minimize*

$$\min_{\mathbf{W}\in\mathcal{W},\mathbf{\Lambda}} ||\mathbf{M} - \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}||_F^2$$

[3]Symmetric NMF: $\min_{\mathbf{W}} ||\mathbf{M} - \mathbf{W}^\top \mathbf{W}||_F^2$

[4]We use congruence here though it is not matrix congruence since $\mathbf{W}$ may not be a square matrix.

and the minimum is defined as the lack-of-fit of $\mathbf{M}$ *regarding* $\mathcal{W}$[5].

Like NMF, it's impossible to ask for the strict uniqueness of the results. Let $P_{\mathbf{\Lambda}}$ be the set of permutation matrices such that $\mathbf{\Pi}^\top \mathbf{\Lambda} \mathbf{\Pi}$ is still upper-triangular. If $(\mathbf{W}, \mathbf{\Lambda})$ is a solution, then $(\mathbf{\Pi}^{-1}\mathbf{D}\mathbf{W}, \mathbf{\Pi}^\top \mathbf{D}^{-1}\mathbf{\Lambda}\mathbf{D}^{-1}\mathbf{\Pi})$ is also a solution where $\mathbf{D}$ is a diagonal matrix with positive elements and $\mathbf{\Pi} \in P_{\mathbf{\Lambda}}$. We state the uniqueness results as follows and the proof is deferred to Appendix C.

Proposition 2.7 (Uniqueness)**.** *If* $|T| \leq |A|$ *and* T *columns of* $\mathbf{W}$ *consist of a permuted positive diagonal matrix, NCT for* $\mathbf{M} = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}$ *is unique in the sense that then for all* $\mathbf{W'}^\top \mathbf{\Lambda'} \mathbf{W'} = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}$*, there exists a positive diagonal matrix* $\mathbf{D}$ *and a* $|T| \times |T|$ *permutation matrix* $\mathbf{\Pi} \in P_{\mathbf{\Lambda}}$ *such that* $\mathbf{W'} = \mathbf{\Pi}^{-1}\mathbf{D}\mathbf{W}$*.*

When we restrict $\mathbf{W}$ to be "**semi-orthogonal**", we obtain a clean format of NCT without searching for optimal $\mathbf{\Lambda}$. $\mathcal{I}$ is the set of all "semi-orthogonal" matrices $\mathbf{W}$ where each column of $\mathbf{W}$ has and only has one non-zero element and $\mathbf{W}\mathbf{W}^\top = \mathbf{I}$. For example, the $\mathbf{W}$ in Example 2.2 can be normalized to a semi-orthogonal matrix. The following lemma follows from the expansion of the Frobenius norm and we defer the proof to Appendix C.

Lemma 2.8 (Semi-orthogonal: minimizing F-norm = maximizing upper-triangular's sum of the square)**.** *For all set of matrices* $\mathcal{W} \subset \mathcal{I}$*,* $\min_{\mathbf{W} \in \mathcal{W}, \mathbf{\Lambda}} ||\mathbf{M} - \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}||_F^2$ *is equivalent to solving* $\max_{\mathbf{W} \in \mathcal{W}} \sum_{i \leq j} (\mathbf{W}\mathbf{M}\mathbf{W}^\top)_{i,j}^2$ *and setting* $\mathbf{\Lambda}$ *as* $\mathrm{Up}(\mathbf{W}\mathbf{M}\mathbf{W}^\top)$*, the upper-triangular area of* $\mathbf{W}\mathbf{M}\mathbf{W}^\top$*.*

2.3 Inferring the thinking hierarchy with answer-prediction joint distribution M

Given $\mathbf{M}$, inferring the thinking hierarchy is equivalent to solving NCT in general. Though we do not have $\mathbf{M}$, later we will show a proxy for $\mathbf{M}$. For simplicity of practical use, we introduce two simple ranking algorithms by employing Lemma 2.8. The ranking algorithm takes $\mathbf{M}$ as input and outputs a linear ordering of answers $\pi : \{1, 2, \cdots, |A|\} \mapsto A$ that maps a ranking position to an answer.

Answer-Ranking Algorithm (Default) $AR(\mathbf{M})$ The algorithm computes

$$\mathbf{\Pi}^* \leftarrow \arg\max_{\mathbf{\Pi} \in \mathcal{P}} \sum_{i \leq j} (\mathbf{\Pi}\mathbf{M}\mathbf{\Pi}^\top)_{i,j}^2$$

where $\mathcal{P}$ is the set of all $|A| \times |A|$ permutation matrices. There is a one to one mapping between each permutation matrix $\mathbf{\Pi}$ and a linear ordering π: $\Pi_{i,\pi(i)} = 1, \forall i$. Therefore, the optimal $\mathbf{\Pi}^*$ leads to an optimal rank over answers directly and the default algorithm can be also represented as

$$\pi^* \leftarrow \arg\max_{\pi} \sum_{i \leq j} M_{\pi(i),\pi(j)}^2.$$

To find the optimal rank, we use a dynamic programming based algorithm (Appendix B) which takes $O(2^{|A|}|A|^2)$. In practice, $|A|$ is usually at most 7 or 8. In our empirical study, the default algorithm takes 91 milliseconds to finish the computation of all 152 questions.

The default algorithm implicitly assumes $|T| = |A|$ and all oracles are deterministic. To allow $|T| < |A|$ and non-deterministic oracles, we introduce a variant that generalizes $\mathcal{P}$ to a subset of semi-orthogonal matrices $\mathcal{I}$. Every $|T| \times |A|$ semi-orthogonal $\mathbf{W}$ indicates a hard clustering. Each

[5] $\mathbf{M} = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}, \mathbf{W} \in \mathcal{W}$ has zero lack-of-fit.

cluster $t \in T$ contains all answers a such that $W_{t,a} > 0$. For example, the $\mathbf{W}$ in Example 2.2 can be normalized to a semi-orthogonal matrix and indicates a hard clustering $\{4\}, \{6, 3\}$. Therefore, the variant algorithm will partition the answers into multiple clusters and assign a hierarchy to the clusters.

Answer-Ranking Algorithm (Variant) $AR^+(\mathbf{M}, \mathcal{W})$ The algorithm computes

$$\mathbf{W}^* \leftarrow \arg\max_{\mathbf{W} \in \mathcal{I}} \sum_{i \le j} (\mathbf{W}\mathbf{M}\mathbf{W}^\top)^2_{i,j}.$$

where $\mathcal{W} \subset \mathcal{I}$. $\mathbf{W}^*$ is normalized such that every row sums to 1. This algorithm does not restrict $|T| = |A|$ and learns the optimal $|T|$.

$\mathbf{W}^* \Rightarrow$ Answer rank The output $\mathbf{W}^*$ indicates a hard clustering of all answers. We rank all answer as follows: for any $i < j$, the answers in cluster i has a higher rank [6] than the answers in cluster j. For all i, for any two answers a, a' in the same cluster i, a is ranked higher than a' if $W^*_{i,a} > W^*_{i,a'}$.

Theoretical justification When $\mathbf{M}$ perfectly fits the model with the restriction that the latent $\mathbf{W}$ is a permutation or a hard clustering, we show that our algorithm finds the thinking hierarchy. Otherwise, our algorithm finds the "closest" solution measured by the Frobenius norm.

Theorem 2.9. *When there exists $\mathbf{\Pi}_0 \in \mathcal{P}$ and non-negative upper-triangular matrix $\mathbf{\Lambda}_0$ such that $\mathbf{M} = \mathbf{\Pi}_0^\top \mathbf{\Lambda}_0 \mathbf{\Pi}_0$, $AR(\mathbf{M})$ finds the thinking hierarchy [7]. In general, $AR(\mathbf{M})$ will output $\mathbf{\Pi}^*$ where $\mathbf{\Pi}^*, \mathbf{\Lambda}^* = \mathrm{Up}(\mathbf{\Pi}^* \mathbf{M} \mathbf{\Pi}^{*\top})$ is a solution to $\arg\min_{\mathbf{\Pi} \in \mathcal{P}, \mathbf{\Lambda}} ||\mathbf{M} - \mathbf{\Pi}^\top \mathbf{\Lambda} \mathbf{\Pi}||_F^2$. The above statement is still valid by replacing $\mathcal{P}$ by $\mathcal{W} \subset \mathcal{I}$ and $AR(\mathbf{M})$ by $AR^+(\mathbf{M}, \mathcal{W})$.*

Proposition 2.7 and Lemma 2.8 almost directly imply the above theorem. We defer the formal proof to Appendix C.

2.4 A proxy for answer-prediction joint distribution M

In practice, we do not have perfect $\mathbf{M}$. We use the following *open-response* paradigm to obtain a proxy for $\mathbf{M}$.

Answer-prediction paradigm the respondents are asked *Q1: What's your answer? Answer:____ Q2: What do you think other people will answer:____.*

In the circle example, the possible feedback can be "answer: 4; prediction: 3", "answer: 3; prediction: 6,9,1"... We collect all answers provided by the respondents [8] and denote the set of them by A. In the circle example, $A = \{1, 2, 3, 4, 6, 9\}$. We also allow respondents to provide no prediction or multiple predictions.

[6] Rank 1 answer is in the highest position.

[7] See definition in the last paragraph in Section 2.1.

[8] In practice, we set a threshold θ and collect answers which are provided by at least θ fraction of the respondents. We allow multiple predictions and also allow people to answer "I do not know". See Section 3 for more details.

Answer-prediction matrix We aggregate the feedback and visualize it by an Answer-Prediction matrix. The Answer-Prediction matrix $\mathbf{A}$ is a $|A| \times |A|$ square matrix where $|A|$ is the number of distinct answers provided by the respondents. Each entry $A_{a,g}, a, g \in A$ is the number of respondents that answer "a" and predict "g".

We will show that with proper assumptions, the answer-prediction matrix $\mathbf{A}$'s expectation is proportional to $\mathbf{M}$. First, for ease of analysis, we assume that each respondent's predictions are i.i.d. samples[9]. Second, since we allow people to optionally provide predictions, we need to additionally assume that the number of predictions each respondent is willing to provide is independent of her type and answer. We state the formal result as follows and the proof is deferred to Appendix C.

Proposition 2.10. *When each respondent's predictions are i.i.d. samples, and the number of predictions she gives is independent of her type and answer, the answer-prediction matrix* $\mathbf{A}$*'s expectation is proportional to* $\mathbf{M}$.

3 Studies

We conduct four studies, study 1 (35 math problems), study 2 (30 Go problems), study 3 (44 general knowledge questions), and study 4 (43 Chinese character pronunciation questions).

Data collection All studies are performed by online questionnaires. We recruit the respondents by an online announcement[10] or from an online platform that is similar to Amazon Mechanical Turk. We get respondents' consent for using and sharing their data for research. Respondents are asked not to search for the answers to the questions or communicate with other people. We allow the respondents to answer "I do not know" for all questions. Except for Go problems, all questionnaires use flat payment. We illustrate the data collection process in detail in Appendix A. We allow respondents to participate in more than one study because our algorithms analyze each question separately and independently.

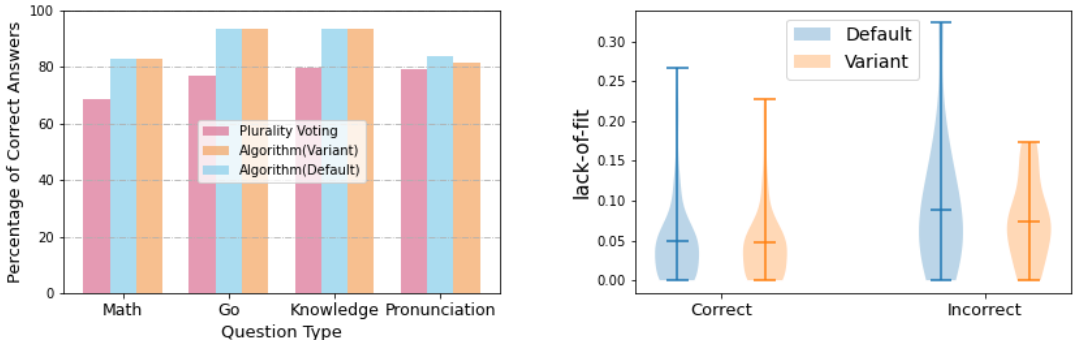
Data processing We merge answers which are the same, like '0.5' and '50%'. We omit the answers that are reported by less than ($\leq$) 3% of respondents or one person. The remaining answers, excluding "I do not know", form the answer set A whose size is $|A|$. We then construct the Answer-Prediction matrix and perform our algorithms. Pseudo-codes are attached in Appendix B. Our algorithms do not require any prior or the respondents' expertise levels.

3.1 Results

We compare our approach to the baseline, the plurality voting, regarding the accuracy of the top-ranking answers. Both of the algorithms beat plurality voting for all studies and the default is slightly better. Among all 152 questions, in 138 questions, the variant algorithm outputs the same hierarchy as the default algorithm. For other questions, the top-ranking type of the variant algorithm may contain more than one answer. The top-ranking answer is the answer supported by more people among all answers in the top-ranking type. In one question the variant is wrong but the

[9]This may not be a very good assumption since i.i.d. samples can repeat but respondents usually do not repeat their predictions. If we do not want this assumption, we can choose to only use the first prediction from each respondent (if there exists) to construct the answer-prediction matrix.

[10]Many are students from top universities in China. See Appendix A for more details.



(a) Accuracy of algorithms

(b) Empirical distribution of lack-of-fit

Figure 3: The results of our experiment

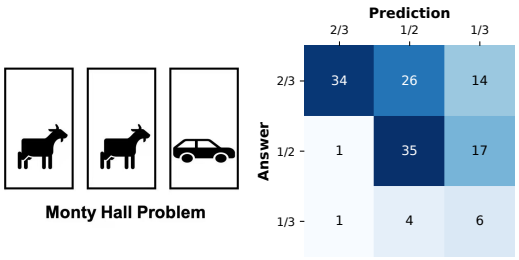
Type	Total	Our algorithm(Default)	Our algorithm(Variant)	Plurality voting
Math	35	**29**	**29**	24
Go	30	**28**	**28**	23
General knowledge	44	**41**	**41**	35
Chinese character	43	**36**	35	34

Table 1: The number of questions our algorithms/baseline are correct.

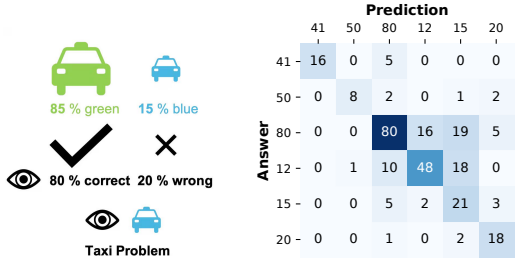
default is correct, the variant algorithm assigns both the correct answer and the incorrect plurality answer to the top-ranking type, thus outputting the incorrect answer as the top-ranking answer.

We also compute the lack-of-fit index (Definition 2.6) of the algorithms and find that the questions the algorithm outputs the correct answer have a smaller lack-of-fit thus fitting the thinking hierarchy model better. Therefore, we can use the lack-of-fit index as a reliability index of the algorithms.

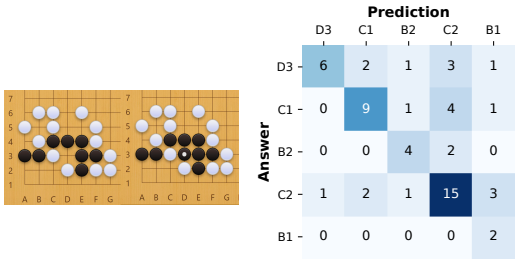
We additionally pick several representative examples for each study (Figure 4) where the matrices are ranked by the default algorithm and the diagonal area modified like Example 1.1. In all of these examples, the plurality is incorrect while our approach is correct. Results of other questions are illustrated at `https://elicitation.info/classroom/1/`. Detailed explanations are illustrated in Appendix D and here we provide some highlights. First, our approach elicits a rich hierarchy. For example, the taxicab problem is borrowed from Kahneman [8] and previous studies show that people usually ignore the base rate and report '80%'. The imagined levels can be 41%→80%. We elicit a much richer level "**41%**→50%→80%→12%→15%→20%". Second, the most sophisticated level may fail to predict the least one. In the Taxicab problem, the correct "41%" supporters successfully predict the common wrong answer "80%". However, they fail to predict the surprisingly wrong answers "12%,20%", which are in contrast successfully predicted by "80%" supporters. Our model is sufficiently general to allow this situation. Third, even for problems (like Go) without famous mistakes, our approach still works. Moreover, in the boundary question, we identify the correct answer without any prior when only 3 respondents are correct.



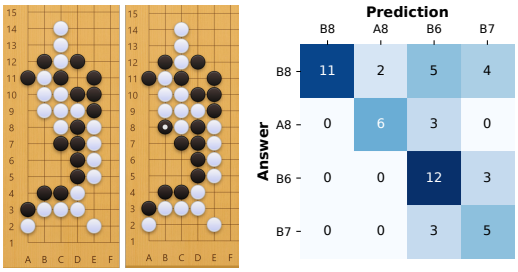
(a) the Monty Hall problem: you can select one closed door of three. A prize, a car, is behind one of the doors. The other two doors hide goats. After you have made your choice, Monty Hall will open one of the remaining doors and show that it does not contain the prize. He then asks you if you would like to switch your choice to the other unopened door. What is the probability to get the prize if you switch?



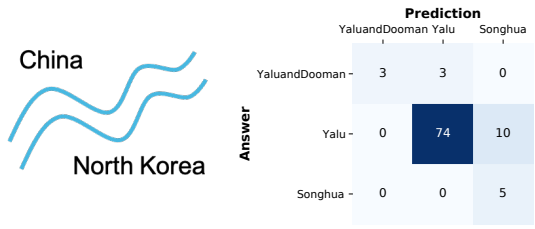
(b) the Taxicab problem: 85% of taxis in this city are green, the others are blue. A witness sees a blue taxi. She is usually correct with probability 80%. What is the probability that the taxi saw by the witness is blue?



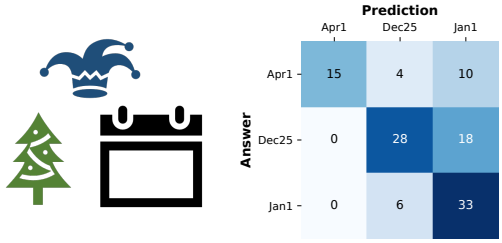
(c) Pick a move for black such that they can be alive.



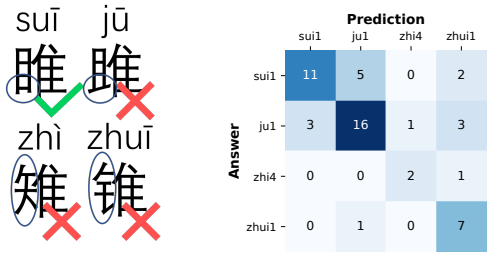
(d) Pick a move for black such that they can be alive by ko.



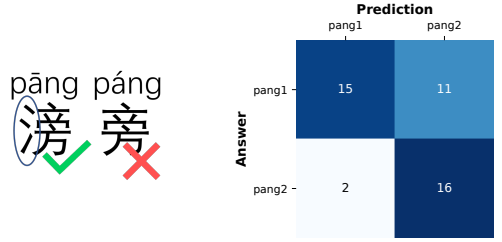
(e) the boundary question: what river forms the boundary between North Korea and China?



(f) the Middle Age New Year question: when was the new year in middle age?



(g) the pronunciation of 睢



(h) the pronunciation of 滂

Figure 4: The ranked answer-prediction matrices

4 Discussion

One future direction is to consider incentives in our paradigm like the literature of information elicitation without verification [13, 18, 5, 22, 9]. We have asked a class of students at Peking University: *why are bar chairs high?* using our paradigm. We cluster the answers by hand. The plurality answer is "the bar counter is high" and our top-ranking answer is "better eye contact with people who stand". Thus, another future diction is to extend our approach to the scenario where people's answers are sentences, where we can apply NLP to cluster them automatically. In summary, we propose the first empirically validated method to learn the thinking hierarchy without any prior in the general problem-solving scenarios. Potentially, our paradigm can be used to make a better decision when we crowd-source opinions in a new field with little prior information. Moreover, when we elicit the crowds' opinions for a policy, with the thinking hierarchy information, it's possible to understand the crowds' opinions better. However, regarding the negative impact, it may be easier to implement a social media manipulation of public opinion with the full thinking hierarchy of the crowds. One interesting future direction is to explore the impact of the thinking hierarchy information.

Acknowledgement

This research is supported by National Natural Science Foundation of China award number 62002001. We would like to thank Xiaotie Deng, Xiaoming Li, Grant Schoenebeck, and David Parkes for their useful suggestions, and all participants of our studies for their time and efforts.

References

[1] Joseph Bertrand. *Calcul des probabilités*. Gauthier-Villars, 1889.

[2] David V Budescu and Eva Chen. Identifying expertise to extract the wisdom of crowds. *Management Science*, 61(2):267–280, 2015.

[3] Colin F Camerer, Teck-Hua Ho, and Juin-Kuan Chong. A cognitive hierarchy model of games. *The Quarterly Journal of Economics*, 119(3):861–898, 2004.

[4] Yi-Chun Chen, Manuel Mueller-Frank, and Mallesh M Pai. The wisdom of the crowd and higher-order beliefs. *arXiv preprint arXiv:2102.02666*, 2021.

[5] Anirban Dasgupta and Arpita Ghosh. Crowdsourced judgement elicitation with endogenous proficiency. In *Proceedings of the 22nd international conference on World Wide Web*, pages 319–330. International World Wide Web Conferences Steering Committee, 2013.

[6] Anirban Dasgupta, Ravi Kumar, and D Sivakumar. Social sampling. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 235–243, 2012.

[7] Hadi Hosseini, Debmalya Mandal, Nisarg Shah, and Kevin Shi. Surprisingly popular voting recovers rankings, surprisingly! *arXiv preprint arXiv:2105.09386*, 2021.

[8] Daniel Kahneman. *Thinking, fast and slow*. Macmillan, 2011.

[9] Yuqing Kong. Dominantly truthful multi-task peer prediction with a constant number of tasks. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2398–2411. SIAM, 2020.

[10] Yuqing Kong and Grant Schoenebeck. Eliciting expertise without verification. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 195–212, 2018.

[11] Y. Liu, J. Wang, and Y. Chen. Surrogate scoring rules. In *EC '20: The 21st ACM Conference on Economics and Computation*, 2020.

[12] Marcellin Martinie, Tom Wilkening, and Piers DL Howe. Using meta-predictions to identify experts in the crowd when past performance is unknown. *Plos one*, 15(4):e0232058, 2020.

[13] N. Miller, P. Resnick, and R. Zeckhauser. Eliciting informative feedback: The peer-prediction method. *Management Science*, pages 1359–1373, 2005.

[14] Martin Müller. Computer go. *Artificial Intelligence*, 134(1):145–179, 2002. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(01)00121-7. URL `https://www.sciencedirect.com/science/article/pii/S0004370201001217`.

[15] Asa Palley and Ville Satopää. Boosting the wisdom of crowds within a single judgment problem: Weighted averaging based on peer predictions. *Available at SSRN 3504286*, 2022.

[16] Asa B Palley and Jack B Soll. Extracting the wisdom of crowds when information is shared. *Management Science*, 65(5):2291–2309, 2019.

[17] Cem Peker. Extracting the collective wisdom in probabilistic judgments. *Available at SSRN 4072021*, 2022.

[18] D. Prelec. A Bayesian Truth Serum for subjective data. *Science*, 306(5695):462–466, 2004.

[19] Dražen Prelec, H Sebastian Seung, and John McCoy. A solution to the single-question crowd wisdom problem. *Nature*, 541(7638):532–535, 2017.

[20] David M Rothschild and Justin Wolfers. Forecasting elections: Voter intentions versus expectations. *Available at SSRN 1884644*, 2011.

[21] Grant Schoenebeck and Biaoshuai Tao. Wisdom of the crowd voting: Truthful aggregation of voter information and preferences. *Advances in Neural Information Processing Systems*, 34: 1872–1883, 2021.

[22] Victor Shnayder, Arpit Agarwal, Rafael Frongillo, and David C Parkes. Informed truthfulness in multi-task peer prediction. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 179–196. ACM, 2016.

[23] Herbert A Simon. Bounded rationality. In *Utility and probability*, pages 15–18. Springer, 1990.

[24] Dale O Stahl. Evolution of smartn players. *Games and Economic Behavior*, 5(4):604–617, 1993.

[25] Dale O Stahl and Paul W Wilson. On players' models of other players: Theory and experimental evidence. *Games and Economic Behavior*, 10(1):218–254, 1995.

[26] James Surowiecki. *The wisdom of crowds*. Anchor, 2005.

[27] Amos Tversky and Daniel Kahneman. Judgment under uncertainty: Heuristics and biases. *science*, 185(4157):1124–1131, 1974.

[28] Susan C Weller. Cultural consensus theory: Applications and frequently asked questions. *Field methods*, 19(4):339–368, 2007.

[29] Tom Wilkening, Marcellin Martinie, and Piers DL Howe. Hidden experts in the crowd: Using meta-predictions to leverage expertise in single-question prediction problems. *Management Science*, 68(1):487–508, 2022.

[30] Pauline R Yu. Allegory, allegoresis, and the classic of poetry. *Harvard Journal of Asiatic Studies*, 43(2):377–412, 1983.

A Data Collection

Study 1: Math problems In total, we assign 35 math problems by 4 online questionnaires. One of them (d) consists of 5 problems. Three of them (a,b,c) consist of 10 problems. Some of these problems are classic interview problems like Monty Hall problem or borrowed from *Thinking, fast and slow*. The other problems are a subset of math Olympiad contest problems for elementary school. The problems cover areas of probability, combinatorics, and geometry.

We recruit the respondents by an online announcement. 76 respondents participate in questionnaire a, 72 respondents participate in b, 28 respondents participate in c, 247 respondents participate in d. Each respondent receives around 0.8 dollars and spends 16 minutes per 10 questions.

Study 2: Life-and-death Go problems We assign 3 questionnaires. Each questionnaire consists of 10 Life-and-death Go problems. The problems are a subset of Life-and-death problems on the quiz site (https://www.101weiqi.com/).

We post an announcement on our public account to recruit respondents who know how to play Go. We prepare a simple sample Life-and-death Go problem and only recruit respondents who answer it correctly. 76 respondents participate in questionnaire a, 39 respondents participate in b, 28 respondents participate in c. Respondents' payments depend on the accuracy of their answers and each respondent receives 3 dollars on average for each study. The average time each respondent spends on each study is about one hour.

Study 3: General knowledge question In total we assign 44 questions by 4 online questionnaires. Questionnaire a consists of 12 questions. Questionnaire b consists of 5 questions. Questionnaire c consists of 10 questions. Questionnaire d consists of 13 questions. Questionnaire e and f both consist of 2 questions. The questions are a subset of the questions of a famous Quiz show in China.

Some respondents are recruited by our online announcement. Some respondents are recruited from an online platform in China (https://www.wjx.cn) which is very similar to Amazon Mechanical turk. 125 respondents participate in questionnaire a. 247 respondents participate in questionnaire b. 98 respondents participate in questionnaire c, 298 respondents participate in questionnaire d, 28 respondents participated in questionnaire e, 35 respondents participated in questionnaire f. Each respondent receives around 0.8 dollars and spends around 4 minutes per 10 questions.

Study 4: Chinese character pronunciation In total we assign 42 questions by 4 online questionnaires a, b, c, d. Two of them (b, c) ask 10 questions. Questionnaire a asks 11 questions. Questionnaire d asks 12 questions. For simplicity of the format, we ask each respondent to label the tone of the pronunciation as a number and attach the number after the answer. For example, $\bar{a} = a1, \acute{a} = a2, \check{a} = a3, \grave{a} = a4$. For the first three questionnaires, we recruit the respondents by an online announcement. The last one is conducted in one class in EECS department at a top university in China. 46 respondents participate in questionnaire a, 63 respondents participate in b, 35 respondents participate in c. 105 respondents participate in d. For each study, each respondent receives around 0.8 dollars and spends 3 minutes on average.

Overall, for math and go study, the respondents who signed up for our studies are highly educated people who are able to understand the terms appearing in the sample questions posted in our announcement. Most of them are from top universities in China. Our results show that even when the majority of these highly educated respondents are wrong, we can identify the correct answer. For the remaining studies, some of the respondents are recruited by the online platform

which is similar to Amazon Mechanical Turk. Our algorithm beats the baseline in these studies as well.

Figure 5: screenshot of instructions given to participants

Instructions given to participants All the questions and instructions are given in Chinese and show as Figure 5. The translation of the upper part instruction is as below. Dear participants, we invite you to complete this questionnaire for a scientific study. We will not collect your name and there is no right or wrong answer to the questions. When filling this questionnaire, please write your true belief and do not search on the internet. If you don't have any clue to a question, you can answer "I don't know". Please complete this questionnaire carefully. Thank you very much. Note that the prediction of other people's answers is to answer specific answers instead of probabilities.

The translation of the example question shown in the lower part of Figure 5 is as below.

1. What is the pronunciation of "陟"?(You can answer "I don't know".The correct way to answer is the characters of pinyin followed by a number representing the tone. For example, the pronunciation of "一" is "yi1", the pronunciation of "黄" is "huang2", the prounciation of "晚" is "wan3", the pronunciation of "大" is "da4".)

2. What do you think other people will answer, to the previous question?(If there are multiple predictions, please use comma to separate them. A possible answer can be: "yi1,er2,san3,si4".)

B Answer-Ranking Algorithm

The default algorithm Our default algorithm aims to find the rank that maximizes the sum of the square of the elements in the upper triangular area of the Answer-Prediction matrix. We can enumerate all possible ranks to find the optimal which requires $O(|A|! * |A|^2)$. Here we use a more efficient dynamic programming algorithm which requires $O(2^{|A|} * |A|^2)$ (e.g. when $|A| = 10$, $2^{|A|} = 1024, |A|! = 3,628,800$). The core observation here is that, the optimal rank of the answer set A can be obtained by the optimal rank of the subset S. Therefore, we can run the dynamic programming by enumerating the subsets in a predetermined order[11]. Figure 6 illustrates the algorithm.

The variant algorithm The following observation helps us pick proper set of $\mathbf{W}$ in the variant algorithm.

[11]The order satisfies that for any set $S_1 \subset S_2$, S_1 is enumerated before S_2.

Observation B.1. *when* $\mathbf{M} = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}$ *and* $\mathbf{W} \in \mathcal{I}$*, for all non-zero elements of* $\mathbf{W}$*,* $\frac{W_{t,a}}{W_{t,a'}} = \frac{p_a}{p_{a'}}$ *where* p_a *is the sum of the* a^{th} *row of* $\mathbf{M}$*.*

Proof. $\mathbf{1}$ denotes a $|A| \times 1$ column vector where all elements are one.

$$\mathbf{p} = \mathbf{M1} = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W1} = \mathbf{W}^\top \mathbf{v}$$

Each row of $\mathbf{W}^\top$ has and only has one non-zero element. Then for any a, a' where $W_{t,a} > 0$ and $W_{t,a'} > 0$, we have $\forall s \neq t, W_{s,a} = 0$ and $W_{s,a'} = 0$. Therefore,

$$\begin{aligned}
\frac{p_a}{p_{a'}} &= \frac{\sum_s W_{s,a} * v_s}{\sum_s W_{s,a'} * v_s} \\
&= \frac{W_{t,a} * v_t + \sum_{s \neq t} 0 * v_s}{W_{t,a'} * v_t + \sum_{s \neq t} 0 * v_s} \\
&= \frac{W_{t,a} * v_t}{W_{t,a'} * v_t}
\end{aligned}$$

Since $\forall a, p_a > 0$, we have $\frac{p_a}{p_{a'}} > 0$ and $v_t \neq 0$. Therefore, we can cancel out v_t to get $\frac{p_a}{p_{a'}} = \frac{W_{t,a}}{W_{t,a'}}$. □

In the model, p_a represents the probability that a respondent answer a. With the above observation, instead of searching over all semi-orthogonal matrices, we can enumerate all possible partitions. In detail, for all $|T| \leq |A|$, we enumerate all possible partitions $\mathbf{b} \in T^{1 \times |A|}$ where each b_a indicates the type answer a belongs to. Each partition $\mathbf{b}$ induces a matrix $\mathbf{W}$ such that for all t, a, $W_{t,a} = C_{t,a} * p_a$. We then normalize $\mathbf{W}$'s rows such that $\mathbf{W}\mathbf{W}^\top = \mathbf{I}$ and pick $\mathbf{W}$ and the optimal order of $\mathbf{W}$'s rows to maximize our objective. Moreover, we reduce the time complexity by using the default algorithm as a building block to pick the optimal order of $\mathbf{W}$'s rows, i.e., the rank of types.

In both algorithms, we also take the percentage of answers $\mathbf{p}$ as input. It is used to construct $\mathcal{W}$ for the variant algorithm and break tie for both algorithms. We state the pseudo-codes as follows.

C Proofs

Proof of Proposition 2.7. We first consider the case where $|T| = |A|$ and $\mathbf{W}$ is a monomial matrix with positive elements. A monomial matrix is a permuted diagonal matrix. Then for all $\mathbf{W'}^\top \mathbf{\Lambda'} \mathbf{W'} = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}$, we have $\mathbf{\Lambda} = (\mathbf{W'}\mathbf{W}^{-1})^\top \mathbf{\Lambda'} (\mathbf{W'}\mathbf{W}^{-1})$.

Note that when $\mathbf{W}$ is a monomial matrix with positive elements, its inverse $\mathbf{W}^{-1}$ is also a monomial matrix with positive elements. Thus $\mathbf{U} = \mathbf{W'}\mathbf{W}^{-1}$ is a non-negative matrix. We will show the following statement. Recall that $P_{\mathbf{\Lambda}}$ is the set of permutation matrices such that $\mathbf{\Pi}^\top \mathbf{\Lambda} \mathbf{\Pi}$ is still upper-triangular.

Claim C.1. *If* $\mathbf{\Lambda} = \mathbf{U}^\top \mathbf{\Lambda'} \mathbf{U}$ *where* $\mathbf{U}$ *is non-negative and both* $\mathbf{\Lambda}, \mathbf{\Lambda'}$ *are non-negative upper-triangular matrices with positive diagonal elements, then* $\mathbf{U} = \mathbf{\Pi}^{-1}\mathbf{D}$ *where* $\mathbf{\Pi} \in P_{\mathbf{\Lambda}}$ *and* $\mathbf{D}$ *is a positive diagonal matrix.*

The above statement implies that $\mathbf{W'} = \mathbf{\Pi}^{-1}\mathbf{D}\mathbf{W}$.

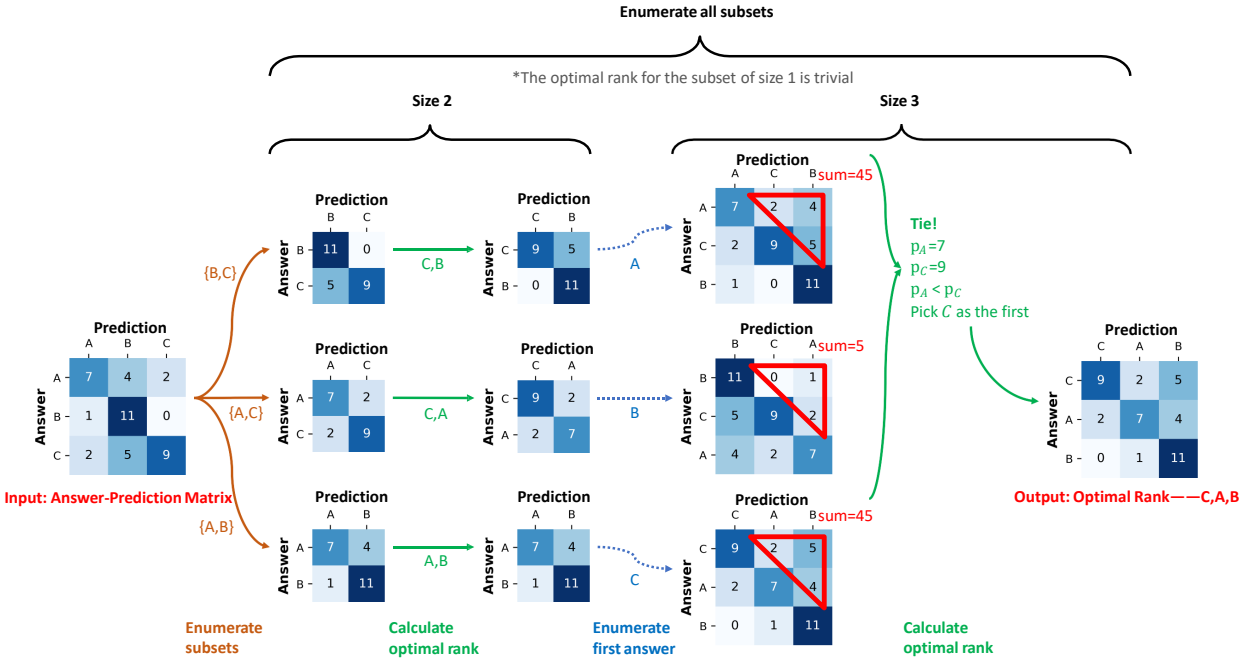


Figure 6: Workflow of the default Answer-Ranking algorithm

Algorithm 1: Answer-Ranking algorithm (default)

Input: Answer-Prediction matrix $\mathbf{M}$, Percentage of answers $\mathbf{p}$, Answer set A
Output: Optimal answer rank π^*
π^*_S is the optimal rank of subset S
Function Up_sum(π) := $\sum_{i<j} M^2_{\pi(i),\pi(j)}$; `// use memoization to reduce the time complexity`
for $S \subseteq A$ **do** `// enumerate all subsets of` A `in a predetermined order`
  initialize π_S;
  for $a_1 \in S$ **do** `// enumerate the first answer in the rank of S`
    $\hat{\pi}(1) = a_1, \hat{\pi}(2:|S|) = \pi^*_{S\setminus\{a_1\}}$; `// attach` a_1 `to the optimal rank of` $S\setminus\{a_1\}$
    $s_1 = \text{Up\_sum}(\hat{\pi})$;
    $s = \text{Up\_sum}(\pi_S)$;
    $a = \pi_S(1)$;
    if $(s_1 > s)$ **then**
      $\pi_S = \hat{\pi}$;
    else if $(s_1 == s) \wedge (p_{a_1} > p_a)$ **then** `// tie-break rule`
      $\pi_S = \hat{\pi}$;
    end
  end
  $\pi^*_S = \pi_S$;
end
$\pi^* = \pi^*_A$;

Algorithm 2: Answer-Ranking algorithm (variant)

Input: Answer-Prediction matrix $\mathbf{M}$, Percentage of answers $\mathbf{p}$, Answer set A
Output: Optimal matrix $\mathbf{W}^*$
$\mathbf{b}$ is a partition of the answer set A, b_a indicates the type answer a belongs to
$\mathbf{B}$ is the set of all possible partitions of answer set A
Initialize $obj^*, \mathbf{W}^*$;
for $\mathbf{b} \in \mathbf{B}$ **do** // enumerate all partitions of A
  Initialize $\hat{\mathbf{W}}$, $\hat{\mathbf{p}}$;
  for $a \in A$ **do**
    $\hat{W}_{b_a,a} = \sqrt{\frac{p_a^2}{\sum_{b_{a'}=b_a} p_{a'}^2}}$; // normalize $\hat{\mathbf{W}}$ such that $\hat{\mathbf{W}}\hat{\mathbf{W}}^\top = \mathbf{I}$
    $\hat{p}_{b_a} = \max(\hat{p}_{b_a}, p_a)$; // will be used to break tie later
  end
  $\hat{\mathbf{\Lambda}} = \hat{\mathbf{W}}\mathbf{M}\hat{\mathbf{W}}^\top$;
  $\pi = AR(\hat{\mathbf{\Lambda}}, \hat{\mathbf{p}})$; // rank the types by the default algorithm
  construct $\mathbf{W}$ such that $\mathbf{w}_i = \hat{\mathbf{w}}_{\pi(i)}$;
  $obj = \sum_{i \le j} (\mathbf{W}\mathbf{M}\mathbf{W}^\top)^2_{i,j}$;
  if $obj > obj^*$ **then**
    $obj^* = obj$;
    $\mathbf{W}^* = \mathbf{W}$;
  end
end

Proof of Claim C.1. We first prove that $\mathbf{U}$ must be a monomial matrix. We use $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{|T|}$ to denote $\mathbf{U}$'s columns. For all i, $\mathbf{u}_i^\top \mathbf{\Lambda}' \mathbf{u}_i > 0$, this implies that $\mathbf{U}$ does not have any column that is zero everywhere. For $i < j$, $\mathbf{u}_i^\top \mathbf{\Lambda}' \mathbf{u}_j = 0$ since $\mathbf{\Lambda}$ is upper-triangular. Therefore, there does not exist t such that both $\mathbf{u}_i(t), \mathbf{u}_j(t) > 0$ since otherwise $\mathbf{u}_i^\top \mathbf{\Lambda}' \mathbf{u}_j > \mathbf{u}_i(t) \Lambda'_{t,t} \mathbf{u}_j(t) > 0$.

Therefore, for each row t of $\mathbf{U}$, there exists at most one non-zero/positive elements because otherwise there must exist $i < j$ such that $\mathbf{u}_i(t), \mathbf{u}_j(t) > 0$ which contradicts with the fact that $\mathbf{u}_i^\top \mathbf{\Lambda}' \mathbf{u}_j = 0$. Thus, there exist at most $|T|$ non-zero elements in $\mathbf{U}$. Given that $\mathbf{U}$ does not have any column that is zero everywhere, $\mathbf{U}$ must be monomial.

Given that $\mathbf{U}$ is monomial, $\mathbf{U}$ can be written as the form of $\mathbf{\Pi}^{-1}\mathbf{D}$. Thus, $\mathbf{\Lambda} = \mathbf{D}\mathbf{\Pi}^{-1\top}\mathbf{\Lambda}'\mathbf{\Pi}^{-1}\mathbf{D}$ which implies that $\mathbf{\Pi} \in P_{\mathbf{\Lambda}}$.

In general, when $|T| \le |A|$ and $|T|$ columns $c_1, c_2, \cdots, c_{|T|}$ of $\mathbf{W}$ consist of a monomial matrix, we use a $|A| \times |T|$ matrix $\mathbf{C}$ such that $\mathbf{WC}$ equals $\mathbf{W}$ at $c_1, c_2, \cdots, c_{|T|}$ and is zero elsewhere. We can construct $\mathbf{C}$ by setting it equal to an identity matrix at $c_1, c_2, \cdots, c_{|T|}$ and zero elsewhere.

When $\mathbf{W}'^\top \mathbf{\Lambda}' \mathbf{W}' = \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}$, we have $\mathbf{C}^\top \mathbf{W}'^\top \mathbf{\Lambda}' \mathbf{W}' \mathbf{C} = \mathbf{C}^\top \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W} \mathbf{C}$. Our previous analysis implies that $\mathbf{W}'\mathbf{C} = \Pi^{-1}\mathbf{D}\mathbf{W}\mathbf{C}$. Thus, $\Pi^{-1}\mathbf{D} = \mathbf{W}'\mathbf{C}(\mathbf{W}\mathbf{C})^{-1}$

Moreover, we also have $\mathbf{C}^\top \mathbf{W}'^\top \mathbf{\Lambda}' \mathbf{W}' = \mathbf{C}^\top \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}$. Thus, there exists a $|T| \times |T|$ matrix $\mathbf{B} = (\mathbf{C}^\top \mathbf{W}'^\top \mathbf{\Lambda}')^{-1} \mathbf{C}^\top \mathbf{W}^\top \mathbf{\Lambda}$ such that $\mathbf{W}' = \mathbf{B}\mathbf{W}$. In such case, $\mathbf{W}'\mathbf{C} = \mathbf{B}\mathbf{W}\mathbf{C}$. Therefore, $\mathbf{B} = \mathbf{W}'\mathbf{C}(\mathbf{W}\mathbf{C})^{-1} = \Pi^{-1}\mathbf{D}$. This implies that $\mathbf{W}' = \mathbf{\Pi}^{-1}\mathbf{D}\mathbf{W}$.

□

□

Proof of Lemma 2.8.

$$\begin{aligned}
||\mathbf{M} - \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}||_F^2 &= \operatorname{Tr}\left((\mathbf{M} - \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W})(\mathbf{M}^\top - \mathbf{W}^\top \mathbf{\Lambda}^\top \mathbf{W})\right) \\
&= ||\mathbf{M}||_F^2 - \operatorname{Tr}(\mathbf{W}^\top \mathbf{\Lambda} \mathbf{W} \mathbf{M}^\top) - \operatorname{Tr}(\mathbf{M} \mathbf{W}^\top \mathbf{\Lambda}^\top \mathbf{W}) + \operatorname{Tr}(\mathbf{W}^\top \mathbf{\Lambda} \mathbf{W} \mathbf{W}^\top \mathbf{\Lambda}^\top \mathbf{W}) \\
&= ||\mathbf{M}||_F^2 - 2\operatorname{Tr}(\mathbf{W}^\top \mathbf{\Lambda} \mathbf{W} \mathbf{M}^\top) + \operatorname{Tr}(\mathbf{W}^\top \mathbf{\Lambda} \mathbf{\Lambda}^\top \mathbf{W}) \\
&\qquad\qquad (\operatorname{Tr}(\mathbf{A}^\top) = \operatorname{Tr}(\mathbf{A}) \text{ and } \mathbf{W}\mathbf{W}^\top = \mathbf{I}) \\
&= ||\mathbf{M}||_F^2 - 2\operatorname{Tr}(\mathbf{W}^\top \mathbf{\Lambda} \mathbf{W} \mathbf{M}^\top) + ||\mathbf{\Lambda}||_F^2 \qquad (\operatorname{Tr}(\mathbf{AB}) = \operatorname{Tr}(\mathbf{BA}))
\end{aligned}$$

Moreover,

$$\begin{aligned}
||\mathbf{\Lambda} - \mathbf{W}\mathbf{M}\mathbf{W}^\top||_F^2 &= \operatorname{Tr}\left((\mathbf{\Lambda} - \mathbf{W}\mathbf{M}\mathbf{W}^\top)(\mathbf{\Lambda}^\top - \mathbf{W}\mathbf{M}^\top\mathbf{W}^\top)\right) \\
&= ||\mathbf{\Lambda}||_F^2 - 2\operatorname{Tr}(\mathbf{\Lambda}\mathbf{W}\mathbf{M}^\top\mathbf{W}^\top) + ||\mathbf{W}\mathbf{M}\mathbf{W}^\top||_F^2
\end{aligned}$$

Note that $\operatorname{Tr}(\mathbf{\Lambda}\mathbf{W}\mathbf{M}^\top\mathbf{W}^\top) = \operatorname{Tr}(\mathbf{W}^\top\mathbf{\Lambda}\mathbf{W}\mathbf{M}^\top)$, we have

$$||\mathbf{M} - \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}||_F^2 = ||\mathbf{\Lambda} - \mathbf{W}\mathbf{M}\mathbf{W}^\top||_F^2 - ||\mathbf{W}\mathbf{M}\mathbf{W}^\top||_F^2 + ||\mathbf{M}||_F^2$$

Therefore,

$$\arg\min_{\mathbf{W},\mathbf{\Lambda}} ||\mathbf{M} - \mathbf{W}^\top \mathbf{\Lambda} \mathbf{W}||_F^2 = \arg\min_{\mathbf{W},\mathbf{\Lambda}} ||\mathbf{\Lambda} - \mathbf{W}\mathbf{M}\mathbf{W}^\top||_F^2 - ||\mathbf{W}\mathbf{M}\mathbf{W}^\top||_F^2$$

The optimal upper-triangular $\mathbf{\Lambda}^*$ should be the upper-triangular part of $\mathbf{W}\mathbf{M}\mathbf{W}^\top$. That is, for all $i \le j$, $\Lambda^*_{ij} = (\mathbf{W}\mathbf{M}\mathbf{W}^\top)_{ij}$. With optimal $\mathbf{\Lambda}^*$, $||\mathbf{\Lambda}^* - \mathbf{W}\mathbf{M}\mathbf{W}^\top||_F^2 - ||\mathbf{W}\mathbf{M}\mathbf{W}^\top||_F^2$ becomes $-\sum_{i\le j}(\mathbf{W}\mathbf{M}\mathbf{W}^\top)_{ij}^2$. Therefore, $\min_{\mathbf{W}\in\mathcal{W},\mathbf{\Lambda}} ||\mathbf{M} - \mathbf{W}^\top\mathbf{\Lambda}\mathbf{W}||_F^2$ is equivalent to solving $\max_{\mathbf{W}\in\mathcal{W}} \sum_{i\le j}(\mathbf{W}\mathbf{M}\mathbf{W}^\top)_{i,j}^2$.

□

Proof of Theorem 2.9. $\mathcal{P}$ is a special case of $\mathcal{W}$. Thus, we can only prove the second part.

Lemma 2.8 directly shows that in general, $AR(\mathbf{M}, \mathcal{W})$ will output the optimal $\mathbf{W}^*$ where $\mathbf{W}^*, \mathbf{\Lambda}^* = \operatorname{Up}(\mathbf{W}^*\mathbf{M}\mathbf{W}^{*\top})$ is a solution to $\arg\min_{\mathbf{W}\in\mathcal{W},\mathbf{\Lambda}} ||\mathbf{M} - \mathbf{W}^\top\mathbf{\Lambda}\mathbf{W}||_F^2$.

When there exists $\mathbf{W}_0, \mathbf{W}_0\mathbf{W}_0^\top = \mathbf{I}$ such that $\mathbf{M} = \mathbf{W}_0^\top\mathbf{\Lambda}_0\mathbf{W}_0$ where $\mathbf{\Lambda}_0$ is a non-negative upper-triangular matrix, $AR(\mathbf{M}, \mathcal{W})$ will output the exact solution. Moreover, when $\mathbf{W}_0\mathbf{W}_0^\top = \mathbf{I}$, the condition of uniqueness also satisfies. Therefore, there exists a positive diagonal matrix $\mathbf{D}$ and a $|T| \times |T|$ permutation matrix $\mathbf{\Pi} \in P_{\mathbf{\Lambda}_0}$ such that $\mathbf{W}^* = \mathbf{\Pi}^{-1}\mathbf{D}\mathbf{W}_0$. After we additionally normalize $\mathbf{W}^*$ to make it row-stochastic, $\mathbf{W}^* = \mathbf{\Pi}^{-1}\mathbf{W}_0$. Thus, the variant algorithm $AR^+(\mathbf{M}, \mathcal{W})$ finds the thinking hierarchy.

□

Proof of Proposition 2.10. Let the number of prediction each respondent gives follow a distribution D where $D(i)$ is the probability that she will give i predictions. Given n respondents, the expected number $A_{a,g}$ will be

$$\mathrm{E}[A_{a,g}] = n * \sum_t p_t \mathbf{w}_t(a) \sum_i D(i) * i * \sum_{t'} p_{t\to t'} \mathbf{w}_{t'}(g) \propto \sum_t p_t \mathbf{w}_t(a) \sum_{t'} p_{t\to t'} \mathbf{w}_{t'}(g) = M_{a,g}$$

□

D Detailed explanations of the empirical results

For study 1, in addition to the circle problem illustrated previously, we additionally pick two famous Bayesian inference problems, the Monty Hall problem [1] and the Taxicab problem [8]. Both problems have a counter-intuitive correct answer. The Monty Hall problem's intuitive answer is "1/2" since the choices seem to be equally good. The correct answer is "2/3". In our study, the most popular answer is the incorrect common sense answer "1/2". The Taxicab problem has two pieces of information: a base rate and a witness's testimony. Without the testimony, the answer will be "15%". Without considering the base rate, the answer will be "80%". By combining the two pieces of information, the correct answer is "41%". In our study, the most popular answer is the "ignoring-base-rate" answer, "80%" since people are usually insensitive to prior probability [27]. Interestingly, our algorithm not only makes the correct answer the top-ranking but also demonstrates levels that are richer than the imagined levels. In the Monty Hall problem, we elicit levels "**2/3**→1/2→1/3" and in the Taxicab problem, we elicit levels "**41%**→50%→80%→12%→15%→20%". We also find that the levels may not have a partial order structure. In the Taxicab problem, the correct "41%" supporters successfully predict the common wrong answer "80%". However, they fail to predict the surprisingly wrong answer "12%,20%", which are in contrast successfully predicted by "80%" supporters. This shows that our approach not only elicits the most valuable answer but also provides a rich thinking hierarchy of people.

Study 2 asks Life-and-death Go problems. Go/Weiqi is a classic board game that origins from ancient China. The core concept of Go game is Life-and-death [14]. A group of stones' status is determined as either being "alive", where they remain on the board indefinitely, or "dead" where the group will have no liberties. The Life-and-death Go problems ask for the move that can kill or save a group of stones. Our study covers a variety of difficulty levels and we pick two interesting representative problems here. Both problems ask for black's move and are illustrated in Figure 4 and correct answers are marked (black moves with white points). For the first problem, 13 people report C2, the plurality answer, and only 5 people report D3, the correct answer. Our algorithm outputs level **D3**→C1→B2→C2→B1. The plurality answer C2 is an aggressive move for black where black can capture a white stone soon. The correct answer D3 is not a good choice at first sight since it allows white stones to survive later. Another popular wrong answer C1 guarantees that the white stones cannot escape later. Moreover, compared to the correct answer D3, both C1 and C2 are more elegant from an artistic view. Other answers B1 and B2 can create the desired pattern in Go game, "eye" (liberties inside a group), in the lower-left corner. For the second problem, our algorithm successfully outputs the correct answer as the top-ranking and elicits level **B8**→A8→C2→B6→B7 while plurality picks B6. In this problem, popular moves A8, B6, B7 seem to be much safer than the correct move B8. However, interestingly, due to a special pattern in the upper part of this problem, adopting a more dangerous move B8 is more beneficial in this situation.

For study 3, we pick two general knowledge questions, the boundary between China&North Korea question and the Middle Age New Year question. Among 82 respondents, 74 respondents report the Yalu River. However, few people know that the Dooman River also makes part of the boundary between North Korea and China. The Songhua River is very famous in the northeastern part of China. Our algorithm elicits the level of "**Yalu & Dooman**→Yalu→Songhua". Another question asks for the date of the Middle Age's new year. The naive answer is Jan 1^{st}. Some people answer the day of Christmas, Dec 25^{th}. the top-ranking answer is Apr 1^{st}[12]. Our algorithm successfully elicits the level "**Apr** 1^{st}→Dec 25^{th} → Jan 1^{st}".

Study 4 is about the pronunciations of Chinese characters. Many Chinese characters are phono-

[12]This is the official answer of a national test in China.

semantic characters and composed of at least two parts. The semantic component indicates the general meaning of the compound character, while the phonetic component suggests the pronunciation of the compound character. We illustrate the results of two questions here. The first question asks the pronunciation of "睢" (sui1)[13]. It is not a commonly used character. Most people are familiar with "雎"(ju1) since this character is used in one of the best-known songs, "Guan ju 关雎", in "Shi Jing" (the Classic of Poetry), which is the first anthology of verse in China [30]. "雎" is very similar to "睢" except for the indexing component. Thus, the most popular answer here is "ju1". Some other respondents mistakenly report "zhi4" ("雉") or "zhui1" ("锥") as they share the same component with "睢". Our algorithm correctly outputs "sui1" as the top-ranking answer and elicits level "**sui1**→ju1→zhi4→zhui1". The second question asks the pronunciation of "滂"(pang1). Like we mentioned before, most Chinese characters are phono-semantic. When we do not know how to pronounce a character, we can answer an easier substituting question: pronouncing the phonetic component. However, sometimes this attempt does not work. Here though "滂" is very commonly used, many people mistakenly pronounce "pang2" ("旁") since "旁" is the phonetic component of "滂". Our algorithm correctly outputs "**pang1**→pang2".

Though we only show the examples where plurality vote fails, for other questions that plurality vote works, our method illustrates a richer result than plurality vote, a hierarchy of the answers. For example, in one question we ask for the age of Li Shimin, Emperor Taizong of Tang, when he took the throne. We also ask the respondents whether the age is less than or greater than 50 years old as a priming effect. Both plurality vote and our algorithm correctly output the true answer, 28 years old. Our algorithm also outputs a hierarchy of all elicited answers "**28**→27→30→35→40→50" and put the answers that use anchoring-and-adjustment heuristics [27] in lower levels.

In our study, people make systematic errors like making a wrong statistical assumption ("1/2" in Monty Hall), ignoring base rate ("80%" in Taxicab), using availability heuristic, i.e., relying on easy-to-search memories ("Yalu River", "Songhua River", "ju1", "Dec 25^{th}, Jan 1^{st}"), answering an easier substituting question (using greedy moves or moves that look elegant, pronouncing the phonetic component), and using anchoring-and-adjustment heuristics ("40, 50" for Li Shimin question) [27]. Importantly, our empirical results show that without any prior, our algorithm labels these errors as less sophisticated thinking types.

[13]The number 1 after sui represents the tone of the pronunciation. There are 4 tones:$\bar{a} = a1, \acute{a} = a2, \check{a} = a3, \grave{a} = a4$.