Better Human Computation Through Principled Voting

Andrew Mao
Harvard University
mao@seas.harvard.edu

Ariel D. Procaccia
Carnegie Mellon University
arielpro@cs.cmu.edu

Yiling Chen
Harvard University
yiling@eecs.harvard.edu

Abstract

Designers of human computation systms often face the need to aggregate noisy information provided by multiple people. While voting is often used for this purpose, the choice of voting method is typically not principled. We conduct extensive experiments on Amazon Mechanical Turk to better understand how different voting rules perform in practice. Our empirical conclusions show that noisy human voting can differ from what popular theoretical models would predict. Our short-term goal is to motivate the design of better human computation systems; our long-term goal is to spark an interaction between researchers in (computational) social choice and human computation.

1 Introduction

Human computation is a fast-growing field that seeks to harness the relative strengths of humans to solve problems that are difficult for computers to solve alone. The field has recently been gaining traction in the AI community as interesting, deep connections between AI and human computation are uncovered (Dai, Mausam, and Weld 2010; Shahaf and Horvitz 2010; Kamar, Hacker, and Horvitz 2012).

Reliable output from human computation generally requires an efficient and accurate way to combine inputs from multiple human agents. For example, *games with a purpose* (von Ahn and Dabbish 2008) produce useful data from many users as they play an enjoyable game, and the advent of *scientific discovery games* (Cooper et al. 2010a; 2010b) harnessed the power of the crowd for scientific research. In EteRNA (`http://eterna.cmu.edu`), players collaborate in folding RNA into its stable shape by submitting different proposals for stable designs. A subset are then synthesized in a laboratory to learn which design is truly the most stable (and to score the players).

Human computation has also expanded into more general tasks with the use of *online labor markets*. *Amazon Mechanical Turk* (MTurk) is perhaps the paradigmatic online labor market. The market connects requesters, who post human intelligence tasks (HITs); and workers, who perform HITs for monetary compensation. HITs often consist of repetitive and subjective tasks that are simple for humans but difficult for computers, such as labeling pictures. By combining different types of tasks into a *workflow*, requesters can achieve more complex objectives.

The input provided by humans via human computation systems is typically quite noisy, and beyond the setting of very simple tasks there is often a need to aggregate information into a collective choice. Naturally, this stage is often crowdsourced as well, often by letting people *vote* over different proposals that were submitted by their peers. For example, in EteRNA thousands of designs are submitted each month, but only a small number of them can be synthesized in the lab. To single out designs for the lab, players vote for their favorites, and the most popular designs are synthesized.

Voting is also frequently used on MTurk. The popular TurKit toolkit (Little et al. 2010b) is essentially a programming language for creating and managing tasks, and in particular provides an implementation of a voting function. This function receives two alternatives and a threshold as input, and posts HITs asking workers to single out their preferred alternative, until the number of votes for one of the alternatives is greater than the given threshold. To implement the common best-3-out-of-5 vote, it is sufficient to elicit three votes, and elicit more only if the first three agents do not all favor the same alternative. The authors give an example where several suggestions for things to do in New York, themselves generated by workers, are sorted using such pairwise comparisons. Little et al. (2010a) also demonstrate how human computation workflows can solve more complex problems using many iterations of voting. However, combining many comparisons from different voters does not yield a straightforward ranking of the alternatives. So what is the best method to construct a such a ranking?

Two research areas are well-equipped for solving this problem. First, mathematicians and economists have for centuries been studying *social choice theory*, the aggregation of individual preferences into a collective decision. In the last two decades, a field which marries computer science and social choice theory—*computational social choice*—has emerged. Most of the work in computational social choice focuses on applying computational paradigms to social choice theory, for example, by studying the compu-

---

A preliminary version of this research was presented at the 4th Workshop on Human Computation (HCOMP '12) as 'Social Choice for Human Computation'. This paper supersedes the prior work and corrects a tie-breaking error in the Kemeny voting rule.

tational complexity of winner determination (Hemaspaandra, Hemaspaandra, and Rothe 1997; Conitzer 2006; Brandt et al. 2008) and manipulation (Faliszewski and Procaccia 2010; Conitzer, Sandholm, and Lang 2007; Procaccia and Rosenschein 2007) in elections. There is little work that applies social choice theory to computer science empirically (see Dwork et al. (2001) for one exception). Second, the field of *utility theory* in economics has produced *discrete choice* or more general *random utility* models (McFadden 1974) that predict choices when agents are presented with different alternatives. Recent AI research has developed deeper connections between utility modeling, social choice, and machine learning (Azari Soufiani, Parkes, and Xia 2012).

Social choice theory and utility theory offer a large variety of models and techniques. Our goal is to give the first principled answer to the question:

How do the prominent vote aggregation methods compare in human computation settings?

Our approach and results. We are interested in settings where there is a true ranking of several alternatives according to quality. Each voter provides us with his own ranking of the alternatives, and our goal is to identify either the entire true ranking or its top alternative.

In the context of social choice, this is a slightly unusual setting because there are no "preferences" over alternatives, only rankings that reflect subjective estimates of the quality of different alternatives. Nevertheless, this setting is a perfect fit with the view of voting rules as *maximum likelihood estimators*. This view was proposed by the Marquis de Condorcet as early as the 18th Century; it was picked up by Young (1988) two centuries later, and more recently studied by AI researchers (Conitzer and Sandholm 2005; Conitzer, Rognlie, and Xia 2009; Procaccia, Reddi, and Shah 2012). The premise is that the ranking provided by each voter is a noisy estimate of the true ranking, which is generated using a known noise model, and a voting rule aggregates such noisy information. An ideal voting rule then outputs the ranking (resp., alternative) that is most likely to be the true ranking (resp., to be the true top alternative).

Models from utility theory are also a natural fit for this setting. In a general random utility model, each agent obtains utility from an alternative according to an underlying deterministic value, corresponding to the true quality, plus an unobserved stochastic error, which influences the observed quality. For any given distribution of stochastic errors, one may design an algorithm for inference of the most likely true quality values, which produces a ranking over alternatives in the same way as a voting rule.

To answer our abovementioned research question, we designed a set of voting experiments and gathered data extensively from human subjects on MTurk. We chose two domains representing human computation tasks with different properties in regard to voter noise, and our core design insight is the ability to reliably adjust the amount of implicit noise with which users perceive a known underlying ground truth. However, as the noise itself is still generated by the voters, we can compare the performance of several methods in realistic conditions. In previous work, Forsythe et al. (1996) have compared voting rules emprically, but not in the information aggregation setting, and Palfrey (2009) has conducted small-scale experiments on aggregating rank orders from voters. Our experiments stand out in two ways: in studying human computation, we collect significantly more data; and we are particularly interested in comparisons at different levels of voter noise.

Based on thousands of empirical rankings from workers on MTurk, we find that human agents produce very different noise than we would expect from theoretical noise models. In particular, we find that ideal ranking methods under common noise models can fare badly with real human voters, while the commonly used and easily implemented plurality rule compares favorably to other more involved methods.

2 Voting Rules and Aggregation Methods

A typical social choice setting has a set of n *voters*, $\mathcal{N} = \{1, 2, \ldots, n\}$ and a set of m *alternatives* (or candidates), $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$. Each voter i has a preference σ_i, which is a total order over $\mathcal{A}$. In other words, each voter ranks the alternatives. Let $\mathcal{L}$ denote the set of all total orders over $\mathcal{A}$. Then, $\sigma_i \in \mathcal{L}$, $\forall i \in \mathcal{N}$. A *preference profile* $\vec{\sigma}$ is a collection of the preferences of the n agents, $\vec{\sigma} \in \mathcal{L}^n$.

Voting Rules. Social choice theorists have developed a large number of voting rules for aggregating individual preferences. Depending on whether the output is a single winning alternative or a preference ranking of all alternatives, a voting rule can correspond to a *social choice function* or a *social welfare function*.[1] A social choice function is a function $C : \mathcal{L}^n \to \mathcal{A}$, while a social welfare function is a function $W : \mathcal{L}^n \to \mathcal{L}$. Note that both functions receive a preference profile as input. Any social welfare function induces a social choice function by selecting the alternative at the first position in the social preference ranking.

In this paper, we consider the following four popular voting rules:

- **Plurality**: Each voter casts a single vote for his most preferred alternative. The alternative that receives the most votes wins. If a ranking is desired, alternatives can be ranked by the number of votes received.
- **Borda**: For each voter who places an alternative at position k in his ranking σ_i, the alternative receives a score of $m - k$. The alternative that receives the highest total score wins. Alternatives are ranked by their total scores.
- **Maximin**: Let $N(a_i, a_j)$ be the number of voters who rank alternative a_i higher than alternative a_j. An alternative i's maximin score is its worst score in a pairwise election, that is $\min_{j: j \neq i} N(a_i, a_j)$. Alternatives are ranked by their maximin scores.
- **Kemeny**: The Kendall tau distance between two preferences σ and σ' is given by

$$K(\sigma, \sigma') = \frac{1}{2} \sum\nolimits_{(a,a') \in \mathcal{A}^2 : a \neq a'} K_{a,a'}(\sigma, \sigma'), \quad (1)$$

[1] In the computational social choice literature, the term "voting rule" sometimes coincides with social choice function, whereas "rank aggregation rule" is equivalent to social welfare function. We do not make this distinction here.

where $K_{a,a'}(\sigma, \sigma')$ is 0 if alternatives a and a' are in the same order in σ and σ' and 1 if they are in the opposite order. Kemeny selects the ranking with the smallest total Kendall tau distance summed over all individual preferences. That is, $W(\vec{\sigma}) = \arg\min_{\pi \in \mathcal{L}} \sum_{i \in \mathcal{N}} K(\pi, \sigma_i)$. Although computing a Kemeny ranking is NP-hard, heuristics are available (Conitzer, Davenport, and Kalagnanam 2006), and it is easily solvable for a few alternatives.

Random Utility. A random utility model can also be used to construct rankings or find a winning alternative. Using the above notation, each alternative a_i has a true value θ_i, and each voter $j \in \mathcal{N}$ observes the alternatives with utilities $\theta_i + \epsilon_{ij}$, where ϵ_{ij} is an stochastic or noise component, for each alternative and each voter. Voters produce preferences based on their observed utilities for each of the alternatives. For example, the Thurstone-Mosteller model (Thurstone 1927; Mosteller 1951) is a random utility model where the ϵ_{ij} are independent and normally distributed with identical variance. We can produce a social welfare function from a random utility model by estimating the true values for the alternatives under the noise in that model, and ranking the alternatives accordingly.

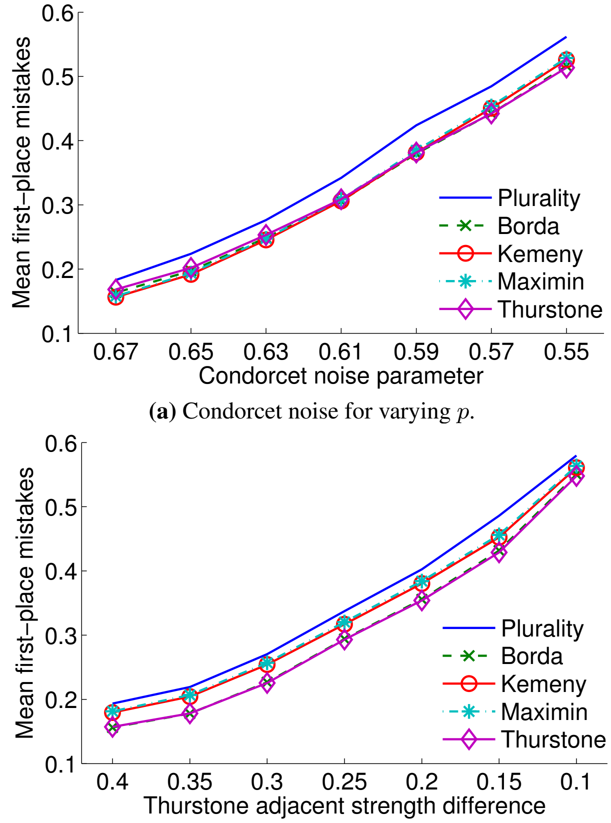
Maximum Likelihood Estimation. Clearly, finding true values for alternatives under a particular random utility model is an example of a maximum likelihood estimation (MLE) problem. However, we can also view voting rules as reconstructing an underlying true ranking of alternatives given noisy information. Using this perspective, there is a body of research seeking to single out voting rules that are MLEs under a model of noisy votes.

More than two centuries ago, Condorcet suggested a natural noise model with an intuitive interpretation: given a true ranking, a voter ranks each pair of alternatives correctly with probability $p > 1/2$.[2] Today this model is also known as the *Mallows* model in the statistics literature. Condorcet solved the case of two alternatives, proving that plurality (known in this case as *majority*) is the maximum likelihood estimator. Moreover, as the number of voters n grows, the probability that plurality will elect the correct alternative increases; it approaches 1 as n approaches infinity (while obvious today, probability theory was in its infancy in the 18th century).

Young (1988) extended Condorcet's solution to the case of more than two alternatives. He showed that, under Condorcet's natural noise model, the voting rule that is most likely to output the correct *ranking* coincides with the Kemeny rule. Young also observed that if p is very close to 1/2 (i.e., when there is a lot of noise), the voting rule that is most likely to select the correct *winner* is Borda. More formally, for every number of voters n and number of alternatives m there is p sufficiently close to $1/2$ such that Borda is an MLE for the top alternative. Finally, Young observed that when p is very close to 1, there are examples where Maximin is the MLE for the top alternative.

3 Comparison Via Synthetic Data

We can confirm the maximum likelihood properties of different ranking methods under theoretical noise models by simulating preference profiles and comparing the performance of voting rules. In addition to simple voting rules, we create a voting rule from the Thurstone-Mosteller model: the model fits all pairwise comparisons over each ranking in a preference profile and estimates the strength parameters using a probit regression. We consider two types of noise: the Condorcet model with noise level p and the Thurstone model with uniformly spaced strength parameters (a fixed $\theta_i - \theta_{i+1}$ for candidates a_i and a_{i+1} in the true ranking). For each model and at each noise level, we generate 100,000 random preference profiles with 4 alternatives and 10 voters, and compute the ranking accuracy as described below.



(a) Condorcet noise for varying p.

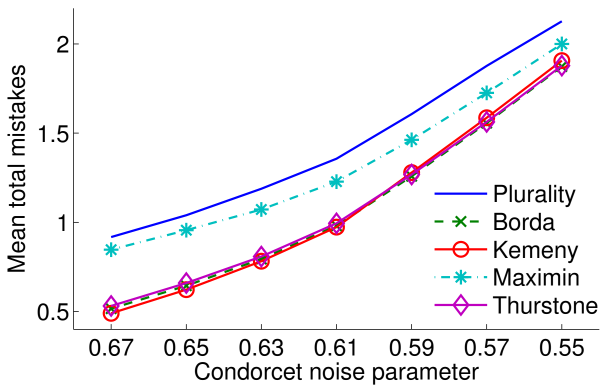
(b) Thurstone noise for varying $(\theta_i - \theta_{i+1})$.

Figure 1: First-place mistakes at different noise levels.

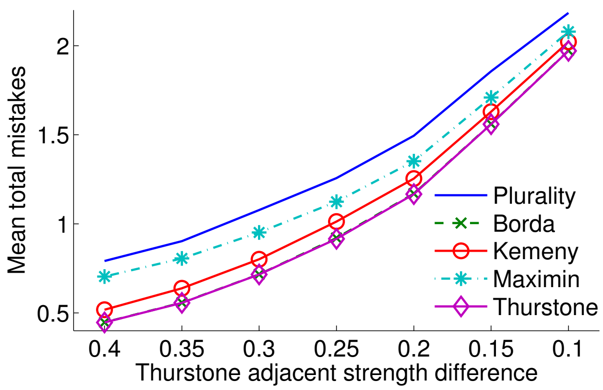
First-place mistakes. To evaluate the voting rules' performance as a social choice function, which elects a single winner, we use the metric of first-place mistakes—the number of preference profiles where a voting rule fails to rank the best alternative at the first position. When more than one alternative ties for first place, we use the expected number of mistakes in random tie-breaking, averaging the number of mistakes across tied alternatives.

Total ranking mistakes. To evaluate how well voting rules perform as social welfare functions, we are interested in how the aggregate ranking compares against the ground truth. The standard, most natural way to measure the difference between two rankings is using the Kendall tau distance, given in Equation (1). The Kendall tau distance between two rankings is simply the total number of pairs on which the

[2]If cyclical preferences are generated, the process is restarted.



(a) Condorcet Noise for varying p.



(b) Thurstone noise for varying $(\theta_i - \theta_{i+1})$.

Figure 2: Total ranking mistakes (Kendall tau distance).

rankings disagree, or the number of pairwise mistakes when compared to the ground truth. Once again, for tied rankings, we average the number of mistakes to compute an expected value for random tie-breaking

Observations. As seen in Figure 1, the plurality rule consistently does worse at winner determination in both types of noise. When comparing overall ranking mistakes in Figure 2, we observe that plurality and Maximin both perform poorly. The Thurstone rule does predictably well with normally distributed noise, but interestingly, Borda performs almost identically. Among simple voting rules, Borda is almost always the best rule in all cases, except for the Condorcet model with $p > 0.6$. where Kemeny performs slightly better. We also point out that while Kemeny is an MLE for the true ranking under the Condorcet model, it doesn't produce the fewest expected mistakes at high levels of noise; instead, Borda performs best here and this is consistent with theory when choosing the winner.

Because of the large amount of data we generated, pairwise differences between ranking methods are almost all highly statistically significant (p-values < 0.0001), except for points that appear to overlap exactly in the figures.

4 Experimental Design

Do the theoretical properties observed above hold up in practice? To answer this, we first identified two voting problems with different characteristics that allowed for both a true ordering and control of ranking noise. We then designed an interface to carefully elicit ranking data from workers on MTurk, and applied the ranking methods described above.

Sliding puzzles. The 8-puzzle consists of a square 3x3 board with tiles numbered from 1 to 8 and an empty space. Starting from any legal board state, one solves the puzzle by sliding the tiles into the empty space to obtain a board state where the numbers are correctly ordered from top to bottom and left to right. Each movement of a single tile counts as one "move", and the general goal is to solve the puzzle in as few moves as possible. An optimal solution to the 8-puzzle game using the fewest number of moves can be found using a search algorithm such as A^*. However, when humans play this game, they will rarely be able to find a solution in the fewest number of moves without significant effort.

Using this idea, we ask users to rank four 8-puzzles by the least number of moves the puzzles are from the solution, from closest to furthest. To collect votes at a certain level of noise, we chose a sequence of numbers, such as $(7, 10, 13, 16)$, and generated a set of four random puzzles solvable in a corresponding number of moves as computed by A^* search. For example, for the above sequence, we would generate one puzzle (approximately uniformly over all such puzzles) that is 7 moves away from the goal, one that is 10 moves away, etc. By fixing the difference between the numbers but varying the overall distance to the goal, we make the puzzles harder or easier to rank relative to each other.

Pictures of dots. The problem of counting pseudo-randomly distributed dots in images has been suggested as a benchmark task for human computation in Horton (2010). Pfeiffer et al. (2012) used the task of comparing such pictures as a proxy for noisy comparisons of items in ranking tasks. We use this latter setting as the basis of voting in our experiments; this task is also easy to explain and requires minimal understanding to complete.

Each voting task involved sorting four pictures from fewest dots to most dots, There were many more dots than could be easily counted; to control the level of noise, we varied the difference in the number of dots among each set of pictures. A larger difference is easier to detect, and therefore less noisy, than a smaller difference.

Comparison of domains. We chose the two domains to represent different types of human computation tasks. When ranking 8-puzzles, there are many ways to approach the problem using heuristics or other solution methods, and we expect (and observe) that workers will expend varying amounts of effort on the task; this is a proxy for tasks where the expertise or quality from workers is very different. However, in the case of counting dots, we took care to ensure that better accuracy is more insensitive to additional effort: the noise from a group of voters will be more homogenous, and this setting gives us an understanding of tasks where input from many voters will be of more equal importance.

4.1 Methodology

Our evaluation tests how different ranking methods fare when applied to noisy collective estimates of a ground truth,

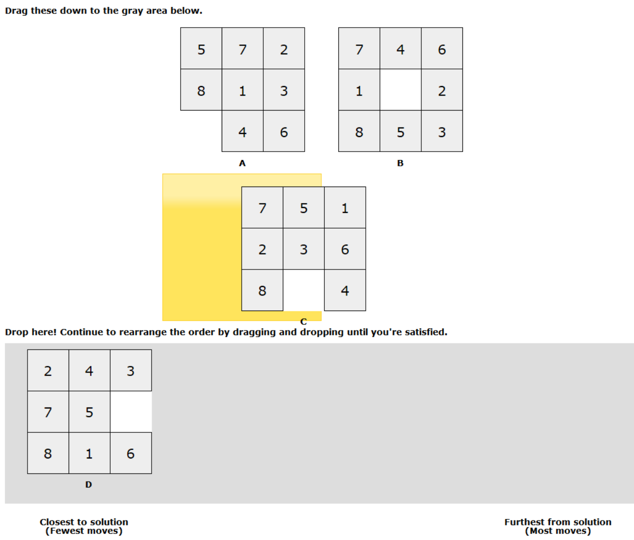


Figure 3: The experiment interface. Voters drag and drop the objects from a square arrangement into a sequence of their preferred order.

comparing them across different levels of noise.

Interface. The core of our interface is an elicitation mechanism for voters to indicate their ranking of alternatives. To collect reliable data for each ranking problem and at different noise levels, we designed our experiment interface carefully, using randomization to reduce behavioral artifacts or potential for bias. Figure 3 shows the interface displaying four 8-puzzles. The objects were presented in randomized order in a square 'starting' grid to workers. Below this was a linear 'target' area with suggestive text anchors at both ends where workers had to order all the alternatives. Objects could be picked up and inserted at any point using drag-and-drop. Moving alternatives to the target area forced workers to make a decision about each one relative to the others; moreover, by randomizing the initial set of objects and arranging them in a square, we removed any bias suggested to low-effort workers by an initial ordering.

We paid \$0.10 for each HIT, which consisted of ranking one set of four objects. Each task began with a basic description of the task, followed by a short quiz to check that users understood how they were comparing puzzles or pictures. Additionally, we enforced a limit of of 5 HITs per user per daily period, and ensured that no user saw the exact same set of objects twice.

Experiment Parameters. After first conducting some initial trials on both data domains, we selected an appropriate level of noise for each set of experiments. For the 8-puzzles, we created puzzles corresponding to four numbers $(d, d{+}3, d{+}6, d{+}9)$, for $d = 5, 7, 9, 11$. For the dot comparisons, we generated pictures containing $(200, 200+x, 200+2x, 200+3x)$ dots for $x = 3, 5, 7, 9$. For each domain and at each level of noise, we generated 40 sets of objects, then collected approximately 20 preference rankings on each set. In other words, we collected 40 *preference profiles* per sequence with 20 voters each, for a total of 3,200 rankings for each type of task.

5 Comparison Via Human Data

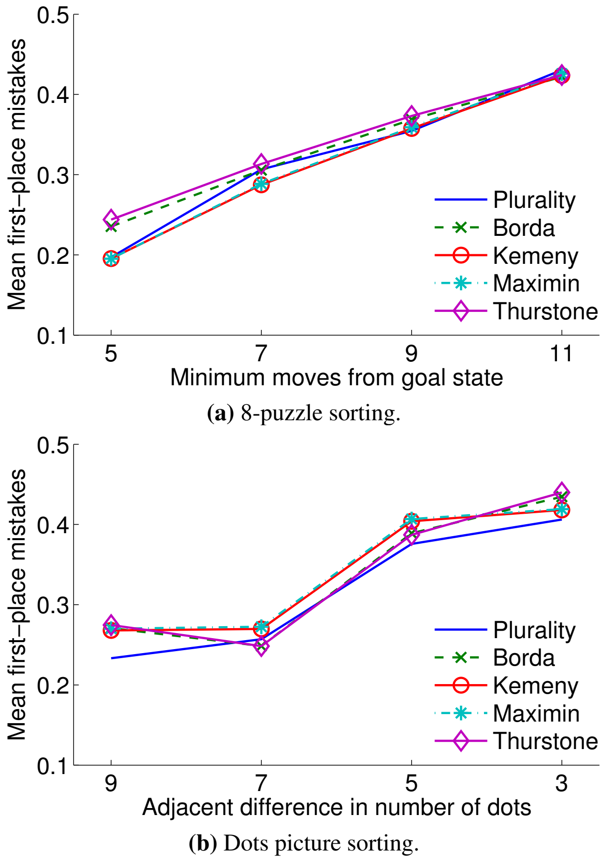
We tested the accuracy of the methods described above in teasing out the correct top alternative and ranking, and computed comparable results to our synthetic data. This data averages the number of mistakes over random subsets of the 40 preference profiles—at each noise level, all voting rules are applied to the same randomly sampled preference profiles consisting of 10 voters. This approach simulates the effect of having fewer voters (and more noise) in aggregation, but also reduces the amount of statistical variance across rules so that they can be compared. Averaging the differences in number of mistakes also creates a normal distribution, allowing for use of a paired t-test for statistical significance.

First-place mistakes. Figure 4 shows the average first-place mistakes for all the ranking methods. The mean number of first-place mistakes for all rules increases as the noise level increases for both ranking problems. This confirms our premise that varying the distance to the goal state in the 8-puzzle or the difference in dots across pictures changes the noisiness of the votes collected. We observe that the Borda rule, predicted to do well in theory, consistently has among the highest number of mistakes; at several points the pairwise difference is significant at the 0.01 level. Meanwhile, the commonly used plurality rule performs much better than in theory, never emerging with the worst score.

Total ranking mistakes. Figure 5 shows the mean Kendall tau distance from the ground truth to the voting rules we tested. Once again, we see that the total number of mistakes for all voting rules rises according to increasing levels of noise. In this case, we also see a distinct difference between plurality and Maximin versus the other voting rules—both have noticeably higher errors when used to construct a ranking (almost all of the differences are significant at the 0.01 level), confirming what we observed in the theoretical models. However, among the other rules, there is no clear winner.

Observations from users. Our entire dataset, including initial exploratory data, consists of 8,529 individual rankings from 1,693 unique voters, including approximately 6,400 rankings from 1,300 unique users in the final evaluation. At \$0.10 per ranking, we collected a large amount of data from a very diverse population in a very economical way.

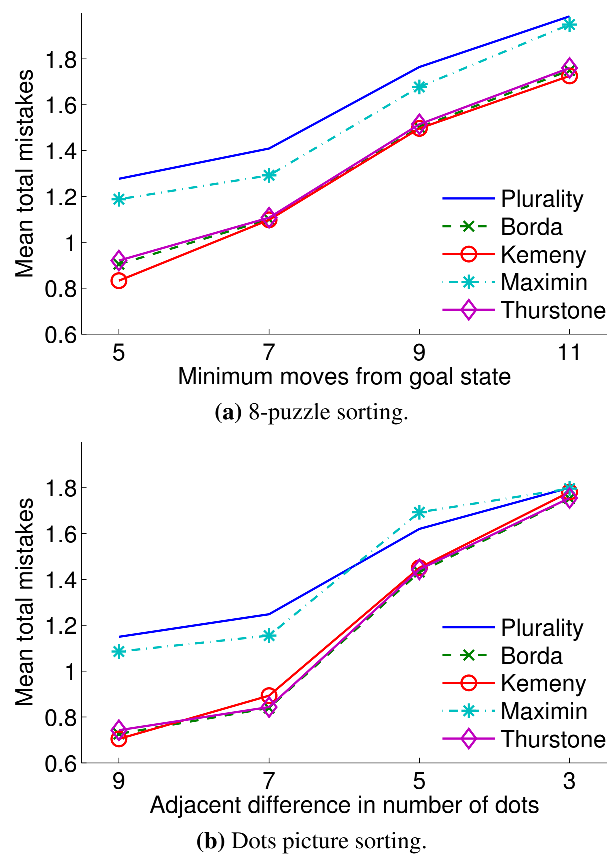
We also asked users about how they approached comparing puzzles. While the individual heuristics of voters do not affect how the voting rules compare, it was interesting to observe different approaches. As expected, the majority of users compared puzzles mentally and did the task quickly, but some also tried to solve the puzzle using pencil and paper, or went even further by constructing a physical 8-puzzle and sliding the pieces around. Others computed what was essentially the Manhattan distance heuristic (an admissible heuristic for A^* search) for each puzzle. A few workers went all the way, wrote code to solve each puzzle, and entered in the minimum number of moves in their comments, even though this was far beyond what we requested. These varying user effort levels are only natural when it comes to human computation settings.



(a) 8-puzzle sorting.

(b) Dots picture sorting.

Figure 4: First-place mistakes at different noise levels.



(a) 8-puzzle sorting.

(b) Dots picture sorting.

Figure 5: Total ranking mistakes (Kendall tau distance).

In our dot comparison tasks, we did not observe any user comments alluding to significant differences in strategies. In particular, voters often commented that they were unable to come up with new ways to solve the task, and desired feedback on their performance. This supports the idea that there was not much beyond simply eyeballing the pictures that could differentiate between them.

6 Discussion

Our results indicate that in realistic human computation settings, there can be significant differences between various methods of aggregating votes. As our results are consistent across two different domains, we believe that they have robust implications for voting in noisy settings. For choosing a ranking, the Borda rule stands out as both simple and accurate: in both theory and experiment, Borda performed as well as Thurstone model—a surprising result given that the latter is a numerically complex probit regression. Our results from real data also support the common use of plurality for the selection of a single alternative, especially since it requires eliciting only one vote instead of a ranking.

Our empirical results (Section 5) stand in contrast to simulations based on the Condorcet and Thurstone noise models (Section 3), where plurality consistently performs poorly. We conclude that the most prominent theoretical noise models are not necessarily good predictors for the performance of different methods on human computation data. It is to be expected that the 8-puzzle data differs from the simulated data, because there the voters are far from being i.i.d. (see Section 5). However, the difference is more surprising when considering the dots data, where we expect workers to be similar in terms of their ability and time investment.

Nevertheless, we believe that theory can play an important role. Indeed, researchers have just begun investigating voting rules that are specifically tailored for human computation and crowdsourcing applications, and provide theoretical guarantees (Goel and Lee 2012; Procaccia, Reddi, and Shah 2012). Moreover, one of our main contributions is our experimental methodology, which allowed us to collect a massive number of high-quality variable-noise votes. This unprecedented dataset[3] facilitates an easy comparison of newly suggested, perhaps tailor-made, voting rules with existing techniques.

In the long run, we hope that our work will spark an interaction between researchers in human computation, computational social choice, and machine learning that will lead to the design of better human computation systems via more principled voting techniques.

Acknowledgments This material is based upon work supported by NSF Grant No. CCF-0915016 and Xerox Foundation.

[3]The dataset consists of voter rankings as well as the puzzle sequences and dot images used to generate them.

References

Azari Soufiani, H.; Parkes, D. C.; and Xia, L. 2012. Random utility theory for social choice: Theory and algorithms. In *Proc. of 26th NIPS*.

Brandt, F.; Fischer, F.; Harrenstein, P.; and Mair, M. 2008. A computational analysis of the Tournament Equilibrium Set. In *Proc. of 23rd AAAI*, 38–43.

Conitzer, V., and Sandholm, T. 2005. Common voting rules as maximum likelihood estimators. In *Proc. of 21st UAI*, 145–152.

Conitzer, V.; Davenport, A.; and Kalagnanam, H. 2006. Improved bounds for computing Kemeny rankings. In *Proc. of 21st AAAI*, 620–626.

Conitzer, V.; Rognlie, M.; and Xia, L. 2009. Preference functions that score rankings and maximum likelihood estimation. In *Proc. of 21st IJCAI*, 109–115.

Conitzer, V.; Sandholm, T.; and Lang, J. 2007. When are elections with few candidates hard to manipulate? *Journal of the ACM* 54(3):1–33.

Conitzer, V. 2006. Computing Slater rankings using similarities among candidates. In *Proc. of 21st AAAI*, 613–619.

Cooper, S.; Khatib, F.; Treuille, A.; Barbero, J.; Lee, J.; Beenen, M.; Leaver-Fay, A.; Baker, D.; and Popović, Z. 2010a. Predicting protein structures with a multiplayer online game. *Nature* 466:756–760.

Cooper, S.; Treuille, A.; Barbero, J.; Leaver-Fay, A.; Tuite, K.; Khatib, F.; Snyder, A. C.; Beenen, M.; Salesin, D.; Baker, D.; and Popović, Z. 2010b. The challenge of designing scientific discovery games. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, 40–47. New York, NY, USA: ACM.

Dai, P.; Mausam; and Weld, D. S. 2010. Decision-theoretic control of crowd-sourced workflows. In *Proc. of 24th AAAI*, 1168–1174.

Dwork, C.; Kumar, R.; Naor, M.; and Sivakumar, D. 2001. Rank aggregation methods for the web. In *Proc. of 10th WWW*, 613–622.

Faliszewski, P., and Procaccia, A. D. 2010. AI's war on manipulation: Are we winning? *AI Magazine* 31(4):53–64.

Forsythe, R.; Rietz, T.; Myerson, R.; and Weber, R. 1996. An experimental study of voting rules and polls in three-candidate elections. *International Journal of Game Theory* 25(3):355–383.

Goel, A., and Lee, D. 2012. Triadic consensus — a randomized algorithm for voting in a crowd. In *Proc. of 8th WINE*, 434–447.

Hemaspaandra, E.; Hemaspaandra, L. A.; and Rothe, J. 1997. Exact analysis of Dodgson elections: Lewis Carroll's 1876 voting system is complete for parallel access to NP. *Journal of the ACM* 44(6):806–825.

Horton, J. J. 2010. The dot-guessing game: A fruit fly for human computation research. Available at SSRN: http://ssrn.com/abstract=1600372 or http://dx.doi.org/10.2139/ssrn.1600372.

Kamar, E.; Hacker, S.; and Horvitz, E. 2012. Combining human and machine intelligence in large-scale crowdsourcing. In *Proc. of 11th AAMAS*.

Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2010a. Exploring iterative and parallel human computation processes. In *Proc. of 2nd HCOMP*.

Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2010b. Turkit: Human computation algorithms on Mechanical Turk. In *Proc. of 23rd UIST*, 57–66.

McFadden, D. 1974. Conditional Logit Analysis of Qualitative Choice Behavior. In Zarembka, P., ed., *Frontiers in econometrics*. New York: Academic Press. 105–142.

Mosteller, F. 1951. Remarks on the method of paired comparisons: I. the least squares solution assuming equal standard deviations and equal correlations. *Psychometrika* 16(1):3–9.

Palfrey, T. R. 2009. Laboratory experiments in political economy. *Annual Review of Political Science* 12:379–388.

Pfeiffer, T.; Gao, X. A.; Mao, A.; Chen, Y.; and Rand, D. G. 2012. Adaptive polling and information aggregation. In *Proc. of 26th AAAI*.

Procaccia, A. D., and Rosenschein, J. S. 2007. Junta distributions and the average-case complexity of manipulating elections. *Journal of Artificial Intelligence Research* 28:157–181.

Procaccia, A. D.; Reddi, S. J.; and Shah, N. 2012. A maximum likelihood approach for selecting sets of alternatives. In *Proc. of 28th UAI*, 695–704.

Shahaf, D., and Horvitz, E. 2010. Generalized task markets for human and machine computation. In *Proc. of 24th AAAI*, 986–993.

Thurstone, L. L. 1927. A law of comparative judgement. *Psychological Review* 34:273–286.

von Ahn, L., and Dabbish, L. 2008. Designing games with a purpose. *Communications of the ACM* 51(8):58–67.

Young, H. P. 1988. Condorcet's theory of voting. *The American Political Science Review* 82(4):1231–1244.