Predicting Human Computation Game Scores with Player Rating Systems

Michael Williams, Anurag Sarkar, Seth Cooper

Predicting Human Computation Game Scores with Player Rating Systems

Michael Williams, Anurag Sarkar, and Seth Cooper(✉)

College of Computer and Information Science
Northeastern University, Boston, USA
{williams.mi, sarkar.an}@husky.neu.edu, scooper@ccs.neu.edu

Abstract. Human computation games aim to apply human skill toward real-world problems through gameplay. Such games may suffer from poor retention, potentially due to the constraints that using pre-existing problems place on game design. Previous work has proposed using player rating systems and matchmaking to balance the difficulty of human computation games, and explored the use of rating systems to predict the outcomes of player attempts at levels. However, these predictions were win/loss, which required setting a score threshold to determine if a player won or lost. This may be undesirable in human computation games, where what scores are possible may be unknown. In this work, we examined the use of rating systems for predicting scores, rather than win/loss, of player attempts at levels. We found that, except in cases with a narrow range of scores and little prior information on player performance, Glicko-2 performs favorably to alternative methods.

Keywords: Human computation games · Player rating systems · Prediction · Elo · Glicko-2

1 Introduction and Background

Human computation games (HCGs) have been shown to provide a unique lens into solving problems that are computationally hard or ill-defined [12, 13]. Some notable examples include *The ESP Game*, which asks users to complete relatively simple image recognition tasks [1], and *Foldit*, which involves relatively complex protein folding problems [4].

One potential upside to leveraging a gaming environment when utilizing human intelligence is the potential to harness the motivational power of games. However, even with exemplar cases such as *Foldit*, human computation games generally have issues engaging and retaining players. Engagement is widely considered a foundational element in a good game. Additionally, the level of engagement experienced by the player can influence how motivated they are to play. The prime factor of engagement is the construct of flow [5], which embodies a range of subjective experiences, but most notably "is the idea that there should be an optimal match between the skills an individual possesses and the challenges presented by an activity" [2, pp.2].

Furthermore, HCGs have several design constraints which limit the extent to which the core task of the game can be edited or modified. Knowing the difficulty of each task within the game beforehand may not be possible, as determining the difficulty of each task by hand circumvents the need to crowdsource the solution. It has been suggested in Cooper et al. [3] that dynamic difficulty adjustment through task ordering may be a logical solution, and that this could be accomplished through the use of player rating systems and matchmaking. They applied player rating systems to an HCG when examining the effect of the bipartiteness of the graph of matches on prediction accuracy of player attempts at levels. To accomplish this, they put in place a somewhat ad-hoc threshold as a "target score", where going beyond the target score counted as a win and failing to do so counted as a loss.

Player rating systems were designed with the intent to give players more fair matches. Several rating systems exist, but the most noteworthy examples include Elo, Glicko-2 and TrueSkill. Elo [7] is a system created by Arpad Elo to rate the relative skill of chess players. His system revolves around a few key assumptions. Mainly, that a player's performance in each match is a normally distributed random variable, and the outcome of a match is the result of a pairwise comparison. Glickman developed the Glicko [9] and Glicko-2 [8] systems, which built upon this model by incorporating additional parameters, notably, a rating deviation parameter and a volatility parameter, which capture the expected rating reliability and fluctuation of a given player. TrueSkill [10] is a rating system developed by Microsoft Research for the purposes of multi-player rating and matchmaking, encompassing both individuals and teams. This is important for their uses and an interesting development because it allows the use of virtually any match configuration (for example, team versus team or free for all).

In this work, we explore generalizing the use of player rating systems to predict outcomes of HCGs from win/loss to continuous scores. As a case study, we used player data from the HCG *Paradox*. Ultimately, when attempting to predict scores, we found that the Glicko-2 player rating system usually outperforms Elo and our baseline measure.

2 Methods

2.1 Data Collection

For this work, we collected data from the puzzle game *Paradox* [6], an HCG that draws on the maximum satisfiability problem (MAX-SAT) to create levels for the players to solve. The game, initially designed to crowdsource formal verification of software, provides various "brushes" for the player to use. These brushes are essentially player guided algorithms to help solve the problems. A player's score is represented as a percentage of satisfied clauses (0%-100%), and the player is given a target score within that range. If a player can complete a level, they have contributed a solution to the underlying MAX-SAT problem. A screenshot of the version of *Paradox* used is given in Figure 1.

Players were recruited to play *Paradox* through Amazon Mechanical Turk (MTurk), where we posted a Human Intelligence Task (HIT). We recruited 50 players, who were paid \$1.50 when they completed the HIT. Upon accepting the HIT, players were given brief instructions about the HIT and game. They then had to complete 9 short tutorial levels meant to introduce gameplay. Data from tutorial levels was not used in our analysis. Players then proceeded to the challenge levels. We selected 33 challenge levels, each of which was either derived from SATLIB Benchmark Problems[1] or randomly generated. These levels were served to the players in random order. Players would not see the same level a second time until they had seen each level at least once. For the challenge levels, players were given a target score of 100% (which is not always necessarily possible). Players were able to skip challenge levels without completing them, and upon skipping 3 levels they could then also exit to complete the HIT. We excluded data from one participant who merely skipped 67 matches without attempting any of them. This brought the participant count to 49 and the total number of matches played to 221.

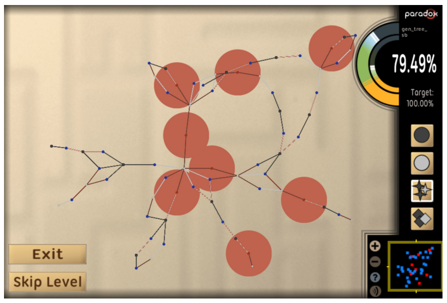

Fig. 1. A screenshot of the game *Paradox* used in this work.

2.2 Rating System Implementation

Our goal was to compare the error of different rating systems when predicting scores achieved by players attempting levels in *Paradox*. Since these systems are conventionally used for the player versus player style games, we have to treat both the players and levels as "players" in the rating system. A match is between a player and a level; players cannot play other players and levels cannot play other levels. The data extracted from the MTurk HIT was played back into the rating systems. We used our own implementation of Elo with a K factor of 24 and the pyglicko2 [11] implementation of Glicko-2. To predict score outcomes using the rating systems, we used expected score of a match based on the ratings of the player and level in the match. For a baseline comparison, we used a simple system that used the average score of all preceding matches to predict the outcome of a match.

To measure the prediction error of each approach, we set up the playback simulation to predict match outcomes *before* playing them back on matches where both the player and level had been in at least some minimum number of matches M. This allowed us to examine the impact the minimum number of matches played on the performance of the rating systems relative to the baseline. It also let us determine how many matches a player and level needs to play before

[1] http://www.cs.ubc.ca/~hoos/SATLIB/benchm.html

Table 1. RMSD error and percentage improvement over baseline. The lowest error in each row is shown in **bold**.

Absolute Score

M	Average	Elo	Glicko-2
0	**0.099**	0.362 (–266%)	0.203 (-105%)
3	0.126	0.239 (-100%)	**0.086** (32%)
6	0.093	0.184 (-158%)	**0.082** (12%)

Relative Score

M	Average	Elo	Glicko-2
0	0.430	0.408 (-5%)	**0.372** (14%)
3	0.508	0.434 (15%)	**0.359** (29%)
6	0.491	0.469 (-4%)	**0.398** (19%)

the rating system starts to outperform the baseline. We used $M = 0$, 3 and 6. If $M = 0$, for example, we predicted the outcome of all matches, and if $M = 3$, we only predicted for matches where the player and level have been in at least 3 previous matches. The simulation is constructed this way because this is the desired use case for a rating system implemented into an HCG. The specific order of the matches played influences the early state of play for both players and levels.

3 Results

Paradox scores can range from 0%-100% but were scaled to a range of 0.0-1.0 for use within the player rating systems. We examined scaling with absolute score (linearly mapping 0% to 0.0 and 100% to 1.0, which is the score shown to the player in game) and relative score (linearly mapping each level's starting % to 0.0 and 100% to 1.0, capturing player improvement over the starting score). The minimum, mean, and maximum absolute scores observed were 0.52, 0.88, and 1.0, respectively, and the minimum, mean, and maximum relative scores were 0.0, 0.53, and 1.0, respectively. The error between observed and predicted values was computed using root mean squared difference (RMSD). Generally speaking, RMSD is a good measurement of accuracy, but specifically accuracy between models measuring the same variable as the scales need to be the same.

Results of our predictions are shown in Table 1. Glicko-2 performs the best in every case except for absolute score with $M = 0$, and improves error over our baseline predictions by up to 32%. As the absolute scores cover a smaller range of possible scores, it is unsurprising that for absolute score predictions the RMSDs are in general lower, and the baseline average score predictions are more accurate.

4 Conclusion

The fact that Glicko-2 outperforms our baseline measure as the system is fed more information about player performance (as M increases) suggests that utilizing a player rating system as a basis for dynamic difficulty adjustment tool could work for HCGs. This is due to the fact that both 3 and 6 minimum matches played seems like a reasonable requisite number of matches played for HCGs before beginning to make predictions. This is especially true if in the long run a system such as Glicko-2 improves player retention. In this sense, the system works to improve player retention but also does its job better the longer they are retained.

Although we found that player rating systems improved prediction error over baseline, it remains to be determined if the accuracy achieved is practically useful. Additionally, the impact of using a matchmaking system based on player rating system score predictions remains to be explored.

Utilizing continuous data as opposed to win/loss unlocks a lot of potential when serving levels to players in HCGs. The surface level improvement is that there is no longer a need to implement a fixed "target score" to allow the rating system to function. Additionally, the precision of predicting and utilizing the score allows for a better determination of what levels are appropriate for which players. For example, if previously the target score was set a threshold of 80%, we can now appropriately differentiate between players who barely beat that target score (i.e. 82%) and players who did far better than the target score (i.e. 98%). Additionally, this may allow for more fine-tuned matchmaking, where each potential player-level match combination has an individualized expected score, and the system recognizes a player who can potentially achieve a new record score on a given level—a very useful feature for HCGs seeking to find new solutions to problems.

Acknowledgements

This work was supported by a Northeastern University TIER 1 grant. This material is based upon work supported by the National Science Foundation under Grant No. 1652537. We would like to thank the University of Washington's Center for Game Science for initial *Paradox* development.

References

1. von Ahn, L., Dabbish, L.: Labeling images with a computer game. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 319–326 (2004)
2. Boyle, E.A., Connolly, T.M., Hainey, T., Boyle, J.M.: Engagement in digital entertainment games: A systematic review. Computers in Human Behavior 28(3), 771–780 (2012)
3. Cooper, S., Deterding, S., Tsapakos, T.: Player rating systems for balancing human computation games: testing the effect of bipartiteness. In: Proceedings of the 1st International Joint Conference of DiGRA and FDG (2016)
4. Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fay, A., Baker, D., Popović, Z., Foldit Players: Predicting protein structures with a multiplayer online game. Nature 466(7307), 756–760 (2010)

5. Csikszentmihalyi, M.: Flow: the psychology of optimal experience. Harper and Row (1990)
6. Dean, D., Gaurino, S., Eusebi, L., Keplinger, A., Pavlik, T., Watro, R., Cammarata, A., Murray, J., McLaughlin, K., Cheng, J., Maddern, T.: Lessons learned in game development for crowdsourced software formal verification. In: Proceedings of the 2015 USENIX Summit on Gaming, Games, and Gamification in Security Education (2015)
7. Elo, A.E.: The rating of chessplayers, past and present. Arco (1978)
8. Glickman, M.E.: Dynamic paired comparison models with stochastic variances. Journal of Applied Statistics 28(6), 673–689 (2001)
9. Glickman, M.E., Jones, A.C.: Rating the chess rating system. Chance 12, 21–28 (1999)
10. Herbrich, R., Minka, T., Graepel, T.: TrueSkill(TM): a Bayesian skill rating system. In: Advances in Neural Information Processing Systems 20. pp. 569–576 (2007)
11. Kirkman, R.: pyglicko2: a Python Implementation of the Glicko-2 algorithm. https://code.google.com/p/pyglicko2/ (2010)
12. Law, E., Ahn, L.v.: Human Computation. Morgan & Claypool (2011)
13. Pe-Than, E.P.P., Goh, D.H.L., Lee, C.S.: A survey and typology of human computation games. In: Proceedings of the 9th International Conference on Information Technology: New Generations. pp. 720–725 (2012)