ISSN No. 0976-5697

Volume 11, Special Issue I, May 2020

International Journal of Advanced Research in Computer Science

CONFERENCE PAPER

Available Online at www.ijarcs.info

AN INTELLIGENT ROCK PAPER SCISSORS BOT IN ANDROID

Mustafa Shihab Assadi*, Natasha Suman, Rachana Jagadish, Pavithra V
School of C&IT, REVA University, Bangalore, India
*mshihab.assadi@gmail.com

Abstract—This project deals with the creation of an intelligent android application that plays the game of Rock, Paper Scissors against the user, learning the user's patterns based on various aspects such as most commonly used inputs and the inputs used in the immediate past. With a custom algorithm, the bot achieves a minimum of a 60% win rate against the average user.

Keywords—rock; paper; scissors; intelligent; pattern recognition

INTRODUCTION

The RPS (Rock Paper Scissors) Bot starts out with an algorithm designed to play with common rps patterns observed in players and aims to predict what object the player will throw, and then counter. As it plays with the same player, it builds patterns based on their unique playing habits and learns their particular quirks and tendencies. The main focus is collecting player data and finding the common outliers among the human players, and then subsequently creating a unique algorithm to counter such opponents, while also building individual player patterns. This particular project focuses on algorithm design, data collection, storage and manipulation, pattern recognition, game theory, and human psychology. As such, it dips its toes into various fields within computer science.

I. GENERAL METHODOLOGY

A. *Mode of implementation*

Android was chosen to be the mode of implementation for ease of use, making it simple for various users to test and play with the app in an easy and accessible format. The user has to simply click on the icon of a rock, paper or pair of scissors, and the relevant move is played.

B. *In-code implementation*

- At first, the computer uses a simple random function to respond to user inputs (hereafter referred to as throws). Very quickly, it builds an idea of the user's commonly used throws, and begins to randomly select one of **three** different algorithms.
- The first algorithm the computer may use is again the simple random function. This is to avoid player cheating, as such, even if one were to know exactly how each algorithm worked, they would not be able to "cheat" the bot.
- The second algorithm used is based on most common player throws. Most players tend to have certain specific throws they use more often than the others, and this algorithm counters those.
- The third algorithm uses a stack-based data structure to remember the user's immediate past few moves, as users also tend to throw similar choices repeatedly over a short interval.

C. *Tracking the effectiveness of the application*

There are two methods of checking how well the algorithms perform their function, one within the app and one through an altered copy of the app. Within the app, a stats screen displays the user's win rate, effectively tracking how well the bot is predicting user throws. Alternatively, a modified copy of the app for debugging displays which algorithm has been chosen on each throw, allowing testers to see how well the given algorithm is working.

II. EXPECTED RESULTS

From testing with n=10 users, we have seen the bot getting an average win-rate of 55% to 70%, well within the desired result of wanting to achieve a 60% win-rate. It is seen that if user throws are purely random, the bot's win-rate drops down to 50%, as expected, since randomness cannot be dealt with. If users do make informed choices like they do when playing with a real opponent, the bot's output is above average.

III. SYSTEM REQUIREMENTS AND DATA FLOW

S.I No.	System Requirements	
	Name	*Min. Version*
1.	Android device	Android 6.0
2.	Bot Application	1.0

Required tools

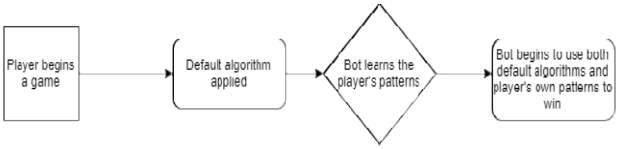
Player begins a game
Default algorithm applied
Bot learns the player's patterns
Bot begins to use both default algorithms and player's own patterns to win

Fig. 1 General flow of data

IV.

C

ONCLUSION

Our rock paper scissors bot thus provides a new implementation of the classic game, with the bot learning as it goes, becoming an ever-evolving challenge to players. With implementation possible in multiple platforms thanks to the use of an algorithm, the bot should be easy for anyone to use, and subsequently watch become smarter with their own eyes. Achieving a high or low win rate then shows how prone the player is to fall into patterns, and how randomly they manage to throw their hand. If the bot has a low win rate, it signifies the player lacking patterns and thus remaining unpredictable, while a high win rate shows the player using the same patterns over and over. Thus, it can also be used to train the ability to refrain from falling into patterns within users.

Expected new applications

- A new approach to player move prediction, with the psychology involved providing insight into a player's fundamental decision making patterns
- A simple "AI" learning a human's behavior over time and learning to respond accordingly

ACKNOWLEDGMENT

Any given task achieved is never the result of efforts of a single individual. There are always a bunch of people who play an instrumental role is leading a task to its completion. Our joy at having successfully finished our mini project work would be incomplete without thanking everyone who helped us out along the way. We would like to express our sense of gratitude to our REVA University for providing us the means of attaining our most cherished goal. We would like to thank our Hon'ble Chancellor, Dr. P. Shyama Raju, Hon'ble Vice-Chancellor, Dr. S. Y. Kulkarni for their immense support towards students to showcase innovative ideas. We cannot express enough thanks to our respected Director, Dr. Sunilkumar S. Manvi for providing us with a highly conducive environment and encouraging the growth and creativity of each and every student. We would also like to offer our sincere gratitude to our Mini Project Coordinators, Prof. Kiran M, Prof. Sailaja Thota, Prof. Surekha Thota and Prof. Ashok K. Patil for the numerous learning opportunities that have been provided. We would like to take this opportunity to express our gratitude to our Mini Project Guide, Prof. Anil Kumar Ambore for continuously supporting and guiding us in our every endeavor as well for taking a keen and active interest in the progress of every phase of our Mini Project. Thank you for providing us with the necessary inputs and suggestions for advancing with our Mini Project work. We deeply appreciate the wise guidance that sir has provided. Finally, we would like to extend our sincere thanks to all the faculty members, staff from School of Computing and Information Technology.

V. REFERENCES

[1] M. Rutledge-Taylor, Robert L. West, "Cognitive modeling versus game theory: why cognition matters", In Proceedings of the Sixth International Conference on Cognitive Modeling, 255-260.Mahwah, NJ: Lawrence Erlbaum, pp 255

[2] Elaine Short, Justin Hart, Michelle Vu, Brian Scassellati,"No fair!!: An interaction with a cheating robot", Applied Soft Computing, 2010, pp. 219

[3] Yuzuru Sato, Eizo Akiyama, and J. Doyne Farmer, "Chaos in Learning a simple two-person game", PNAS,vol. 99., Issue No 7., 2002, pp 4748-4751