11-16-2010

Artificial neural network for bot detection system in MMOGs

Kusno Prasetya
Bond University, Kusno_Prasetya@bond.edu.au

Zheng da Wu
Bond University, Zheng_Da_Wu@bond.edu.au

Artificial Neural Network for Bot Detection System in MMOGs

Kusno Prasetya
School of Information Technology
Bond University
Gold Coast, Australia
Email: kupraset@bond.edu.au

Wu Zheng da
School of Information Technology
Bond University
Gold Coast, Australia
Email: zwu@bond.edu.au

***Abstract*—Cheating is one of the biggest and constant problems in MMOGs. Games with high frequency of cheating will surely lose its appeal to genuine players who want to play the game. This is the reason why game provider these days put cheating prevention as one of the top priorities. Bot is just one way of cheating, but very efficient one. There are various methods to prevent cheating using bot. In this paper, we examine the potential of Artificial Neural Network (ANN) to detect and recognize bot from human players. We start with the assumption that one bot always acts in the similar pattern in gameplay. Meanwhile, it is much more rarer to see 2 players with similar gameplay pattern. The result of our experiment supports our initial hypothesis with the potential for future research in order to get better results.**

***Index Terms*—Cheating, Bot, multiplayer online games, Artificial Neural Network.**

I. Introduction

Multiplayer Game is one type of games that allow player to play with other players in the same game. These days, there are multiplayer games that attract a large number of players and they are called Massively Multiplayer Online Games (MMOG). One of the common problems in MMOGs is the use of bot to cheat. Bot is a popular way of cheating in MMOGs which unlike other types of cheating, it does not involve the attempt to change game state where it violates game mechanics. Cheating using bot is more difficult to detect compared to other types of cheating because it does not make illegal changes according to most of game mechanics.

This paper summarizes our research to detect the usage of bot in MMOG using Artificial Neural Network (ANN). In our research, we experimented with 2 types of recognition tasks: bot movement and bot action patterns. We found that given sufficient training data, ANN could theoretically recognize a bot in MMOG by checking its pattern of movement or action. In practice, our bot detection system can suggest potential cheating players in MMOGs and leave further judgment to human administrator.

II. Related Work

There are many ways to cheat in MMOGs as explained and classified by [1], which includes the use of bot. Commonly, a player needs to run the bot program along with the game. It is possible to counter the cheat by detecting if such programs are running in the player's PC or game console. Punkbuster [2] is an example of such countermeasures program.

Another popular method to detect bot players is CAPTCHA. However, it is possible to implement automatic character recognition within the bot, making CAPTCHA inefficient. Research by [3], [4] suggest variations in CAPTCHA system in order to improve it and making CAPTCHA more difficult for bot to evade or trick. Other popular methods of detecting bot in MMOGs include: [5]–[9] where they perform statistical operations to detect bot presence. Meanwhile, the use of ANN in cheat detection is still in early stage with one recent work by [10]. They propose the use of ANN to detect cheating in player's movement speed with very low false-positive recognition rate.

Our mechanism is highly influenced by [11], where they use ANN for dead reckoning. Their ANN takes previous coordinates as input and produces predicted coordinate as output. Meanwhile, our system takes previous coordinates as input and produces an index that signifies the possibility of bot usage.

III. ANN for Cheating Detection

To obtain the input of ANN, first we look into the category of bot based on the activity. We have 2 categories of bot activity: moving (Type 1) and action (Type 2). For Type 1, we choose to use spatial properties of a player such as its coordinates (x, y, z) and movement speed (v). For action type, we observe a general pattern of action performed by a bot. For example, a bot in the game World of Warcraft may follow the same pattern for its actions. One way to illustrate those action sequences is by using Finite State Machine (FSM). Each state represents an action where state 1 is the idle state. State 2 is when a bot is engaging the enemy. State 3 when a bot is fleeing from an enemy and state 4 is when a bot is moving around in the game world.

We obtain our training data for Type 1 by extracting them from recorded bot movement of the game Quake 2. With the recorded movement data, we get the sequence of bot movement coordinates. The features we extract are the yaw (ϕ), pitch (θ) and bot movement speed (v). For Type 2, we simulated player input data based on the state machine described above. We programmed our bot action simulator

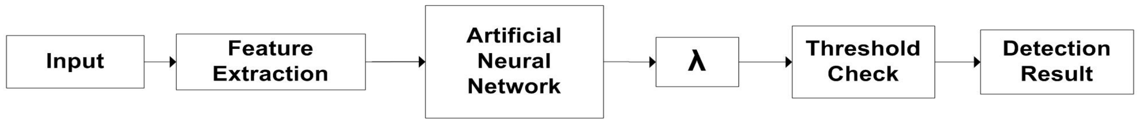

Fig. 1. General Processes of Bot Detection System

to produce random sequence with probability for state 1-2-3-4 as 15%-20%-5%-60%. The result of simulator is a vector a_t^2, where $a_t^2 \in x = 1..4$. For the configuration of ANN, we created 2 different configurations based on trial and error method. Both ANN uses Multi-Layer Perceptron architecture and Resilient Propagation as their learning algorithm. The output for ANN Type 1 (λ_1) and ANN Type 2 (λ_2) are what we call confidence level for each type and the values are between 0 to 1. The value 0 signifies big error and it indicates a large possibility that the current player is a human. Consequently, when the output value is closer to 1, it means the recognition error is low and that indicates that the current player is a bot. Figure 1 describes the general process of our bot detection system.

IV. Experiment Result

For ANN Type 1, we experimented with 2 different bots from the game Quake 2: Gladiator bot and Reaper bot. We used the movement data of Gladiator bot for ANN training and movement data of Reaper bot for cross-reference. From our experiment, we receive the following results:

- With valid training data and valid input, our ANN Type 1 produces output above 0.8 throughout game session regardless of the fluctuation of λ_1.
- When fully trained using data produced by Gladiator bot, λ_1 is always below 0.27 when we use the input generated by Reaper bot.
- The accuracy of both types degrades significantly during simulated packet loss. When there is only 1 packet loss, it does not affect the output. However, if packet loss is frequent, the disruption in input sequence causes the pattern to break.

V. Conclusion

Based on our experiment result, we conclude that ANN has the potential to aid game administrator to detect bot. It strictly needs valid and proper training and input data in order to perform in adequate level which severely restricts its application. Furthermore, latency and packet loss could affect the accuracy significantly. As a result, ANN can be used to provide additional filter for cheater, not as a standalone solution. Finally, ANN is still an ongoing research field and we have an ongoing research to find an efficient ANN construction method that can be used against various bots. Also, with the experiment of Type 2, we are convinced that ANN could be used to detect other type of cheating. However, different type of cheating requires completely different ANN model.

References

[1] S. D. Webb and S. Soh, "Cheating in networked computer games: a review," in *DIMEA '07: Proceedings of the 2nd international conference on Digital interactive media in entertainment and arts*. New York, NY, USA: ACM, 2007, pp. 105–112.

[2] E. B. Inc., "Punkbuster online countermeasures," http://www.evenbalance.com/.

[3] R. V. Yampolskiy and V. Govindaraju, "Embedded noninteractive continuous bot detection," *Comput. Entertain.*, vol. 5, no. 4, pp. 1–11, 2007.

[4] P. Golle and N. Ducheneaut, "Keeping bots out of online games," in *ACE '05: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*. New York, NY, USA: ACM, 2005, pp. 262–265.

[5] T. Schluessler, S. Goglin, and E. Johnson, "Is a bot at the controls?: Detecting input data attacks," in *NetGames '07: Proceedings of the 6th ACM SIGCOMM workshop on Network and system support for games*. New York, NY, USA: ACM, 2007, pp. 1–6.

[6] S. Gianvecchio, Z. Wu, M. Xie, and H. Wang, "Battle of botcraft: fighting bots in online games with human observational proofs," in *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2009, pp. 256–268.

[7] R. Thawonmas, Y. Kashifuji, and K.-T. Chen, "Detection of mmorpg bots based on behavior analysis," in *ACE '08: Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*. New York, NY, USA: ACM, 2008, pp. 91–94.

[8] S. Mitterhofer, C. Kruegel, E. Kirda, and C. Platzer, "Server-side bot detection in massively multiplayer online games," *IEEE Security and Privacy*, vol. 7, pp. 29–36, 2009.

[9] K.-T. Chen, A. Liao, H.-K. K. Pao, and H.-H. Chu, "Game bot detection based on avatar trajectory," in *ICEC '08: Proceedings of the 7th International Conference on Entertainment Computing*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 94–105.

[10] O. B. Gaspareto, D. A. C. Barone, and A. M. Schneider, "Neural networks applied to speed cheating detection in online computer games," *International Conference on Natural Computation*, vol. 4, pp. 526–529, 2008.

[11] A. Mccoy, T. Ward, S. Mcloone, and D. Delaney, "Multistep-ahead neural-network predictors for network traffic reduction in distributed interactive applications," *ACM Trans. Model. Comput. Simul.*, vol. 17, no. 4, p. 16, 2007.