Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 2, No. 1, 2022

Applications of AI in game plug-ins detection

Cheng Fang

Zhejiang Ivy international Academy, Hangzhou, 310000, China
fangc742@gmail.com

Abstract: With the development of the machine learning and computer games, more and more artificial intelligence technologies are applied in the game field. From IBM's Deep Blue's victory over chess king Kasparov in 1997 to AlphaGo's victory over Go world champion Lee Sedol in 2016, AI technology has played a pivotal role, thus making artificial intelligence famous. However, the applications of AI against the game plug-ins have been rarely reported. Thus, this paper will introduce some AI algorithms used in game plug-ins detection and some application cases.

Keywords: Artificial Intelligence; Video games; Plug-ins detection.

1. Introduction

Artificial intelligence (AI), also known as machine intelligence, refers to machines or systems made by humans that can exhibit a certain level of intelligence. In 1920, a British scientist named Babbage developed the world's first "computing machine", which marked the beginning of computer hardware and is now considered as the predecessor of AI. The development and application of electronic computers have made the realization of AI a reality. AI can imitate human behavior, and as people explore AI more and more deeply and the computing power of computers becomes more and more powerful, AI also becomes more "intelligent". The development of AI has gone through a very long history. Game AI is the product of the development of AI technology in the game field, and it has always been at the forefront of its development, and is considered to be the "fruit fly" of AI. Wolfenstein 3D was released back in 1992, and its soldiers also had a basic form of AI.

In the context of the rapid development of the mobile Internet, the improvement of big data mining capabilities and storage capabilities, the leap in computer computing speed, and the accumulation of deep learning algorithms have ushered in new opportunities for game artificial intelligence technology. From IBM's Deep Blue's victory over chess king Kasparov in 1997 to AlphaGo's victory over Go world champion Lee Sedol in 2016, AI technology has played a pivotal role, thus making artificial intelligence famous. In addition, Texas Hold'em AI [2,3] and Mahjong AI in the incomplete information environment have achieved high-profile achievements. For real-time games, King of Glory AI, Dota2 AI and StarCraft II AI have all reached or surpassed the level of top human professional players. Craig Reynolds proposed a theory of simulating the behavior of a group of animals aiming at simulating bird flight, namely dispersion, alignment, and cohesion. The A* pathfinding algorithm is the most popular pathfinding algorithm today. The A* algorithm has the characteristics of real-time, stability, and intelligence, and is simple and easy to use. Therefore, A* Pathfinding algorithm is popular in most of the pathfinding in the game uses. For example, the sports games "NBA2K" series, "FIFA" series, and the shootout game "Battlefield" series, the cumbersome intelligent pathfinding technology in the game is all done using the A* pathfinding algorithm. However, the applications of AI against the game plug-ins have been rarely reported. Thus, this paper will introduce some AI algorithms used in game plug-ins detection and some application cases.

2. AI algorithms

2.1. Neural networks

Inspired by the human brain, the neural network consists of a number of neurons which can receive the perceptual information of the nerve endings. The neuron structure is shown in Figure 1b where x is the neuron input, w is the connection weight between each neuron, b is the threshold, f is the activation function, and o is the output value, which can be expressed by the Equation 1.

$$o = f(\mathbf{wx} + \mathbf{b}) \tag{1}$$

Multiple neurons connecting with each other form a neural network, as shown in Figure 1a. A complete neural network usually consists of three layers, namely the input layer, the hidden layer and the output layer. The number of hidden layers is not less than one, while the input layer and output layer generally have only one layer.

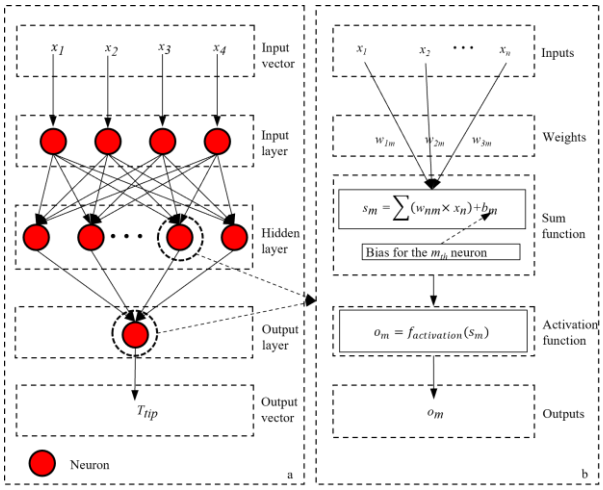

Figure 1. The structure of (a) neural network and (b) neuron

The training process of the neural network model is to adjust the weights of connections between neurons and the biases of neurons iteratively until the specific requirements are satisfied. The diagram in Figure 2 illustrates the flow diagram of the training ANN model

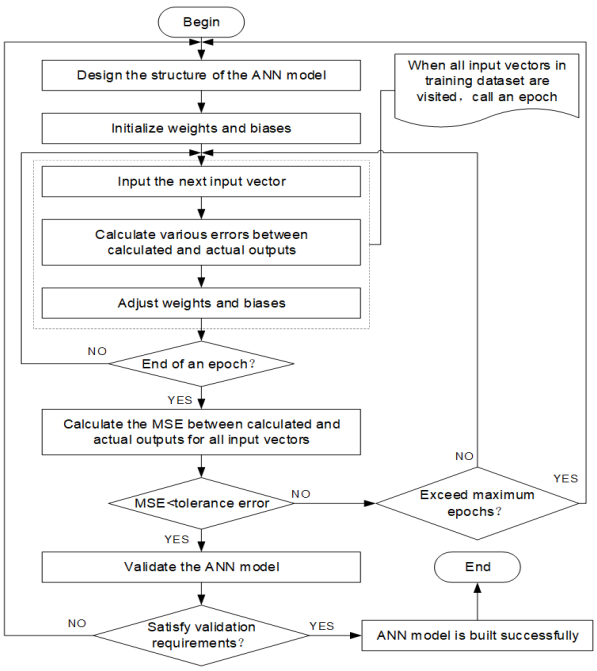

Figure 2. The training process of the neural networks

2.2. Support vector machine

The aim of the support vector machine (SVM) is to maximize the interval between different categories, so that the classification of SVM has higher reliability and generalization ability. Support vectors are those data points close to the boundary during the SVM classification process. The training aim of a support vector machine is to find an optimal hyperplane that correctly splits positive and negative samples. The optimal hyperplane wx + b = 0 can be obtained by doing a nonlinear mapping. When the sample set is linearly separable, the determination of w and b can be obtained based on Equation 2.

$$\begin{cases} \min \frac{1}{2}\|w\|^2 \\ s.t\ y(\mathbf{wx}+\mathbf{b}) \geq 1 \end{cases} \tag{2}$$

When the sample set is not linearly separable, a slack variable ξi > 0 and a penalty factor C should be added, which represent the tolerance for outliers. Thus, Equation 2 becomes a convex quadratic programming problem as shown in Equation 3.

$$\begin{cases} \min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \quad i=1,2,3,...,N \\ s.t\ y(\mathbf{wx}+\mathbf{b}) \geq 1 \end{cases} \tag{3}$$

Solving the linear inseparable problem, the usual method is to apply the kernel function which aims at mapping a low-dimensional sample set to a high-dimensional space. Commonly used kernel functions include Gaussian radial basis kernel function, polynomial kernel function and so on. Introducing the kernel function k(xi, xj) and the Lagrange multiplier αi, Equation 3 can be transformed into the following form as shown in Equation 4.

$$\begin{cases} \min \frac{1}{2}\sum_{i=1}^{N} y_i y_j y \alpha_i \alpha_j k(x_i, x_j) - \sum_{i=1}^{N}\alpha_i \\ s.t\ \sum_{i=1}^{N}\alpha_i y_j = 0, 0 \leq \alpha_i \leq C, \quad i=1,2,3,...,N \end{cases} \tag{4}$$

2.3. Decision tree

Decision tree is a classification method, belonging to supervised learnings. The aim of decision tree training is to choose the optimal division attribute. The optimal division attribute, for binary classification, is to make the divided samples belong to the same category, that is, the attribute with the highest "purity". The "information entropy" is used to measure the purity of features. If the proportion of the k_{th} class samples in the current sample set D is p_k (k = 1, 2, 3, ···, $|K|$), and K is the total number of classes. Then the information entropy of the sample set is calculated as **Equation 5**.

$$Ent(D) = -\sum_{k=1}^{K} p_k \log p_k \tag{5}$$

The smaller the value of Ent(D) indicates the higher the purity of D. Assuming that the discrete attribute a has V possible values $\{a_1, a_2, \cdots, a_V\}$, if the feature a is used to divide the dataset D, V branch nodes will be generated. The V_{th} node contains the total number of samples in data set D whose value is denoted as D_v. Therefore, the information entropy can be calculated according to the above equation of information entropy. Considering that the number of samples contained in different branch nodes is different, a weight $|D_v|/|D|$ is assigned to the branch node. The branch node with more samples will have greater impacts on the information entropy. The "information gain" obtained by dividing the sample set D by feature a can be calculated via **Equation 6**.

$$Gain(D,a) = Ent(D) - \sum_{v=1}^{V}\left(\left|D^v\right|/\left|D\right| Ent(D^v)\right) \tag{6}$$

In general, the larger the information gain indicates the corresponding feature used for classification works effectively.

3. Applications of AI in game plug-ins detection

3.1. keyboard and mouse simulation plug-ins detection

The keyboard and mouse simulation plug-in are a plug-in that replaces the player's operation by simulating the manual operation of the mouse and keyboard through software. The more popular plug-ins are: button wizard, mouse linker, simple game and other plug-in software. Online game developers generally use two methods to detect such plug-ins, one is client-side detection, and the other is user behavior analysis and detection. Client software detection mainly includes detection of the process, window name, interface name, machine code and software feature code of the program running on the player's computer. To give a simple example, some plug-in software is made by hackers for sale. If the sales volume is relatively high and the plug-in is used more often, then some feature codes of the software will be added to the blacklist. This kind of plug-in will prompt illegal, but if you put the plug-in code into other programming languages for editing and then open it, it will not prompt illegal plug-in. This is also one of the pain points of game developers. If the game prompts illegal because of code editing tools, this It violates the copyright regulations formulated by the state, so such methods can only curb the proliferation of plug-ins and cannot completely ban plug-ins. Therefore, the most popular method at present is also the method used by game developers: user behavior analysis and detection, mainly in the following

points:

1)Analyze the mouse click frequency and movement frequency. If there is a long-term click frequency that is the same or similar, or the click frequency is too fast, it will be added to the blacklist.

2)Judging the difference value of game characters, there is a big gap between the behavior of characters in general simulation software and the behavior of players. This gap can be seen by security personnel at a glance, but game developers can't have so many artificial intelligences to all players. Therefore, the game developer will set a range value for the behavior of the game character, that is, the critical point. If the critical value is exceeded, it will be added to the blacklist or the pop-up window will be illegal, and the account will be forced offline.

3)Monitoring result data, the game company strictly monitors the key data in the game, uploads the results of the key data at large time intervals, and performs critical value detection on the data uploaded to the server. It is judged that it is illegal to use analog plug-ins.

Generally, these three points are combined and analyzed based on machine learning methods to outcome more detection results.

3.2. Game data modification detection

In the history of online game anti-cheat, there used to be a very popular anti-cheat system N-Protect, which was widely used in online games at that time. The game running environment, whether there is a plug-in running, whether the client is complete, and whether there is a plug-in to modify the client memory data. However, the plug-in detection method itself has defects, because the anti-plug-in system N-Protect itself exists on the client computer of the game. The computer is owned by the player. How can we ensure that the plug-in program will not be modified and deceived? The client If it can be modified, then the anti-plug-in program can also be modified, so after the anti-plug-in system has been put into operation for a period of time, it was broken by domestic hackers, and finally the system ended in failure. There are the following difficulties in detecting game plug-ins:

1)It is impossible to completely prevent the game client from being modified. Once the game client is in the hands of players and hackers, it is impossible to prevent hackers from modifying. Even with legal constraints, it is impossible to effectively supervise their personal behavior in the current social situation

2)The inspection of the game client is easy to be deceived. When the detection system checks the integrity of the client, the plug-in maker can completely forge a safe process, put reasonable codes in it, and let the detection system detect the fake data, so that No matter how the client is modified, the detection system will not find it.

3)The plug-in detection system cannot correctly transmit the detected results to the server, and the plug-in producer finds the data of its detection results. Even if the detection system has detected illegal software, cheaters can tamper with the inspection results and become detection for legitimate data branches, no matter what plug-ins are detected by the detection system, they cannot send correct detection results to the server.

The machine learning algorithms, which have strong power to reveal the complex relation based on big data analysis, are believed to solve the above issues well.

4. Conclusion

This paper firstly introduces the necessity of AI applied in game plug-ins detection. The common AI algorithms including neural network, support vector machine, decision tree have strong power to reveal the complex relations which are hard to be solved by the traditional analytical mathematics based on sufficient data analysis. The common plug-ins detection technologies include mouse and keyboard simulation detection technology and game data modification detection technology, but these two technologies have their own limitations. Fortunately, it is believed that these limitations can be solved by the machine learning methods because their powerful detection ability. With the development of the AI, more and more complex game plug-ins can be detected so that the game environment will become better.

References

[1] Zhang Jiajia,Qian Tao. The application of artificial intelligence in mobile entertainment and game industry from AlphaGo[J]. Guangdong Economy，2018（1）：48-55.

[2] BROWN N, SANDHOLM T. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. Science, 2018, 359(6374): 418 – 424

[3] BROWN N, SANDHOLM T. Superhuman AI for multi-player poker. Science, 2019, 365(6456): 885 – 890.

[4] LI J, KOYAMADA S, YE Q, et al. Suphx: Mastering mahjong with deep reinforcement learning. arXiv Preprint. Arxiv: 2003.13590, 2020.

[5] YE D, LIU Z, SUN M, et al. Mastering complex control in MOBA games with deep reinforcement learning. Proceedings of the 34th AAAI Conference on Artificial Intelligence. New York, USA: AAAI, 2020: 6672 – 6679.

[6] BERNER C, BROCKMAN G, CHAN B, et al. Dota 2 with large scale deep reinforcement learning. arXiv Preprint. Arxiv: 1912.06680, 2019.

[7] VINYALS O, BABUSCHKIN I, CZARNECKI W, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature, 2019, 575(7782): 350 – 354.