IJARSCT
ISSN (Online) 2581-9429
ISSN (Print) 2581-XXXX
nternational Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 11, Issue 2, November 2020
Impact Factor: 4.819

Use of Artificial Intelligence and Machine Learning in Games

Nachiket Jadhav[1], Aniket Matodkar[2], Anish Mandhare[3] and Sujata Bhairnallykar[4]
Students, Department of Computer Engineering[1,2,3]
Head, Department of Computer Engineering[4]
Saraswati College of Engineering, Kharghar, Navi Mumbai
Affiliation of Mumbai University, Navi Mumbai, India
nachiketjadhav0606@gmail.com[1], aniket.matodkar@gmail.com[2]
anishmandhare7738@gmail.com[3] and bsujata999@gmail.com4

Abstract: *With modern video games surpassing every set of expectations in terms of graphics, game play, mechanics and hardware support, Artificial Intelligence in video games has also come a long way, from when it was first implemented in 1951. Although every set of games has an AI unique to itself, many of the algorithms are now developed such that they can be implemented in various games without any major changes in coding. But this could lead to the players exploiting AI in a single game to break the other games as well. Though this could be easily fixed by changing some minor fragments of algorithms, it would very well be an efficient way of developing complex AI for many games at once. This paper focuses on providing a cost-efficient way to implement AI algorithms that would benefit most of the upcoming and future games that will depend on AI to make themselves more dynamic to the players. This is done by taking the examples of various AI algorithms implemented in games like Pacman, Dota2, Tom Clancy's- The Division and many more.*

Keywords: Artificial Intelligence, Games, Machine Learning, Players.

I. INTRODUCTION

Artificial Intelligence in games is still a vague concept. Many of the next generation games are focusing on how to improve the AI of the game to provide a dynamic and reactive environment for the player base. There have been drastic improvements in video games in the past ten years. With increasing complexity of games, they have also gotten much more interesting and engaging.

The term "AI game" is used to refer to a wide set of algorithms that also include techniques of control theory, robotics, computer graphics and computing generally, and so video game AI may often not be a "true AI" to the extent that such techniques do not necessarily facilitate computer-based learning or other standard criteria, constituting only an "automated calculation" or a predetermined and limited group of responses to a predetermined and limited set of entries.

Even if AI in games has evolved over the past years, developers are hesitant to build a complex AI for their system as they fear losing control of the player experience that they had intended. A good AI is not that can handle the most complex situations, but a one that enhances the player experience makes the game more interactive. [6]

II. EARLY STAGES OF ARTIFICIAL INTELLIGENCE IN GAMES

A) Nim

An early example of AI was Nim's computer game made in 1951 and published in 1952. Although it is an advanced technology in the year it was created, 20 years before Pong, the game took the form of a relatively small box and was able to regularly win games even against the highly skilled players of the game. The Nimrod computer by Ferranti was used in the game of Nim to demonstrate its mathematical capabilities. Nim is a two-player game where players remove one to three objects from a collection of objects alternately. In one type of variation, the player that removes the last

IJARSCT
ISSN (Online) 2581-9429
ISSN (Print) 2581-XXXX
nternational Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 11, Issue 2, November 2020
Impact Factor: 4.819

object is the winner. 480 vacuum tubes were used in nimrod, the precursor to transistors, for processing and display of data .[4]

While Nimrod was custom-designed for playing Nim, Ferranti believed that constructing a machine to play a relatively hard game would mean solving complex problems. Although Nimrod could play Nim efficiently, there was no way for a hardwired set of logic to execute some complex functions. Some of the earliest implementations of machine learning was implemented by Arthur Samuel of IBM in 1956 with the invention of Alpha-Beta Pruning.[6]

B) Alpha Beta Pruning

Alpha-beta size is an altered version of the minimax algorithm. As we saw in the minimax search algorithm, the number of game states it needs to look at is exponential in the depth of the tree. Therefore, there is a technique whereby, without verifying each knot of the game tree, we can calculate the correct minimum decision, and this technique is called pruning. Its implicates two alpha and beta threshold parameters for future expansion, so it is called alpha-beta pruning. The effectiveness of alpha-beta size depends heavily on the order in which each node is looked at. The most important aspect of alpha-beta pruning is the move order.

C) Results of the Early Artificial Intelligence

Forty years later, the alpha-beta pruning method returned after it was implemented in the game of chess. It defeated Garry Kasparov, a grandmaster in chess.The searches of game-state were also performed by Deep Blue, pruned with alpha-beta pruning, but in parallel to increase the speed to identify the computer's next move. Although early games lacked any AI in them, they relied on the concept of state machines to remain unpredictable for the players and had a basic implementation of path finding algorithms in them to keep the player engaged.

III. PATHFINDING AI IN GAMES

To understand how AI is implemented in video games, we are going to look at pathfinding algorithms, which are the base functionalities that every game needs to have perfected. AI in games is mainly used to move bots in the world. In order to move a bot from one point to another, a path must be generated that will lead to the destination in the fastest time and takes less computation time as well so that resources of the hardware are efficiently consumed. Moreover, the pathfinding AI also has to consider all the obstacles in the way and reroute the path dynamically if an obstacle appears in the path after the route is calculated. An example of a collection of constraints can be to seek out the shortest path to require an associate degree agent from its current position to the target position. Pathfinding systems generally use the pre-processed representations of the virtual world as their search area.[4] Pathfinding is approached in two main ways in games, the approaches being undirected and directed. They are as follows:

A) Undirected

In a graph, the edges that have no direction are considered as undirected. In case that all the edges are undirected, the graph is termed as directed graph.[4] An example of an undirected graph is illustrated in figure 1.

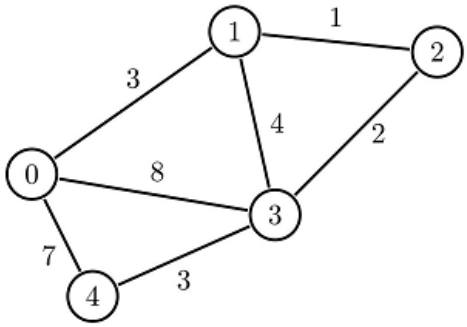

Figure 1: An undirected graph with 5 vertices and edge weights.[6]

IJARSCT
ISSN (Online) 2581-9429
ISSN (Print) 2581-XXXX
nternational Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 11, Issue 2, November 2020
Impact Factor: 4.819

There are two main approaches to find the smallest path in undirected graphs, and are used as they improve the efficiency of the pathfinding without having any major performance issues on the hardware. These approaches are Breadth-first search and Depth-first search respectively. The time complexity of Breadth-first search is:

$$O(|V| + |E|)$$

as every vertex and edge will be explored in the worst case. |V| represents the number of vertices and |E| represents the number of edges in the graph. Note that O(|E|) may vary between O(1) and O(|V|^{2}), depending on how sparse the input graph is.

Dijkstra's Algorithm is another way to find the shortest path from a starting node to the end node in a weighted graph. The weights can represent the distance to be tralleed by the player in the virtual world and this algorithm would determine the shortest path to the destination. The graph has the following:

a. vertices or nodes, denoted by u or v;

b. weighted edges that connect two nodes: (u,v) denotes an edge, and w(u,v)denotes the weight.

This is done by initializing three values:

a. dist, an array of that stores the distance from the source node s, to each node in the graph, and is initialized with a minimum value of 0; and for all other nodes v, dist(v) = ∞.This is done at the start as a result of because the formula return, the dist from the supply to every node v within the graph are going to be recalculated and finalized once the shortest distance to v is found

b. Q, a queue of all nodes within the graph. At the top of the algorithm's progress, the queue is going to be empty.

c. S, an empty set, to indicate the nodes the algorithm has visited. At the end of the algorithm's run, S contains all the nodes of the graph.

The time complexity of Dijkstra's Algorithm is given by

$$O(|E| + |V| \log |V|)$$

which makes it much more efficient than BFS and DFS methods which take much higher time in pathfinding algorithms.

B) Directed

Directed graphs are the graphs that have edges with direction. The edges indicate a one-way relationship such that each edge can only be traversed in one direction. Typically the cost in game maps is distance between the nodes. This will result in algorithms finding a path to the destination, but itty not be the optimal path i.e. the shortest path. The main strategies for directed pathfinding algorithms are:

a. Uniform Cost Search: In this, the smallest weight path next to the active node is always selected to reach the destination.

b. Heuristic search: This estimates the cost from every next node to the destination and cuts the search cost considerably when compared to uniform cost search.

There are two algorithms implemented using the directed graph and they are Dijkstra's algorithm and A* algorithm. A* is another directed algorithm used to find the shortest path. It assesses the best path, and even backtraces its path if necessary. This means that A* will not only find a path between two points but it will find the shortest path if one exists and do so relatively quickly.[4]

The pseudo-code is as follows:

1. Let P = starting point.
2. Assign the values of f,g and h to P.
3. Add P to the Open list. It serves as an initial element in the open list.
4. Let B be the best node from the Open list (i.e. the node that has the lowest f-value).
    a. If B is the destination, then return – a path has been found.
    b. If the Open list is empty, then exit– path is not available
5. Let C be a valid node connected to B.

IJARSCT
ISSN (Online) 2581-9429
ISSN (Print) 2581-XXXX
nternational Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 11, Issue 2, November 2020
Impact Factor: 4.819

a. Assign the values of f,g and h to C.
b. Check if C is on the Open or Closed list.
i. If so, check if there is a new efficient path(i.e. has a lower f-value).
1. If there is, update the path.
ii. Else, add C to the Open list.
c. Repeat step 5 for all valid children in B.
6. Repeat step 4.

IV. Machine Learning

Machine learning is a way for the developers to train the pathfinding AI in ways that it did not encounter during the training or development phase of training the AI. This ensures that when an unexpected situation is placed for the AI, it comes up with an efficient solution and it does not cause any complications for the players and users. Machine learning algorithms are divided into three steps of optimisation, training and limitations. Artificial Neural Networks and Genetic Algorithms are two machine learning approaches that are used in current games. [2]

A) Artificial Neural Networks

These are computational algorithms which are intended to simulate the behaviors of neurons, where multiple neurons are connected which can compute values from inputs. The learning of models in ANN's are classified in two types of Supervised and Unsupervised Learnings. While Supervised learning requires human intervention during multiple stages of development, unsupervised learning can train the model without any human support once it is set up. [4]

B) Genetic Algorithms

The genetic algorithm functions by filling a system with organisms, each with randomly chosen genes that control how the organism behaves in the system. Each organism is scanned with the fitness algorithm to find the two fittest organisms of the system. These then contribute their genes to their offspring, which is then added into the population. The fitness function depends on the problem, but in any case, it is a function which takes one individual as an input and returns a real number as output. The technique of the genetic algorithm tries to directly imitate the evolutionary process, by making selections and crosses with randomized operations of crossing and mutation on populations of programs, algorithms or sets of parameters. Genetic algorithms and genetic programming have yielded truly remarkable results over the last few years.

a) Reinforcement Learnging

Reinforcing learning is a reformulation of the overall AI problem. An agent in an environment receives precepts, maps some of them to positive or negative uses, and then must decide what actions to perform. To avoid reconsidering the whole AI and addressing the principles of reinforcement learning, we need to look at how the learning task may vary:

a. The environment can be accessible or inaccessible. States that can be identified with percepts are said to be in accessible environments, whereas in an inaccessible environment, the agent must maintain some internal state to try to keep track of the environment.

b. The agent begins with knowledge of the environment and the effects of its actions; or it will have to learn this model as well as utility information.[2]

V. How Machine Learning Can Help In Pathfinding Problems

The problem with a static algorithm of pathfinding has many problems, which include

a. If the world constantly changes as the player progresses further into the game, the AI can become deprecated and at one point, would become unable to find optimal paths for the player, resulting in various bugs and would ultimately push the player away.

IJARSCT
ISSN (Online) 2581-9429
ISSN (Print) 2581-XXXX
nternational Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 11, Issue 2, November 2020
Impact Factor: 4.819

b. This would lead to wastage of hardware resources the player has, which can heavily affect the hardwares that runs on maximum performance to run the game.

c. This ultimately would lead to multiple crashes and heavy memory wastages.

Learning algorithms help generalize the above problems by allowing the AI to make dynamic decisions and adapt to the changing dynamic world. It is done with the help of API's (Application Programming Interface), which are collection of specific methods that are prescribed by the programmer writing program that it can make requests to. This allows reusability of code that the programmers can reuse in future games which are more complex and extend on the currently developed functions. Training the model helps the AI use information from the network and generate effective solutions. This helps the AI make reasonable decisions when it is presented with a new situation that it did not encounter during training. This also solves the problem of resource consumption as neural networks do not need heavy resource usage during implementation and handle real time inputs along with data processing in the backgrounds, thus making it much more viable to implement.[3]

VI. CONCLUSION

The applications of Artificial Intelligence are limitless and when correctly implemented, can be used to make dynamic and interactive environments for the players. But development of such a dynamic AI requires a lot of time and resources and it is the main reason why developers are hesitant to experiment with it. One game that did use unsupervised machine learning was Black & White where the human player was able to train their own creature. A neural network was wont to manage the creature however whereas the human player was able to train the creature through reinforcement learning, it absolutely was tightly controlled to avoid all unpredictable/unrealistic behaviour. So it seems that until the game developers are shown proof that machine learning can overcome the limitations of standard approaches they will avoid it.

From the above algorithms mentioned, we can successfully decipher that Dijkstra's Algorithm for shortest path takes much less time than the rest of the above algorithms and can be used for efficient use of resources, both hardware and memory.

REFERENCES

[1]. Ross Graham, Hugh McCabe, Stephen Sheridan " Pathfinding in computer games"

[2]. John DeNero and Dan Klein "Teaching Introductory Artificial Intelligence with Pac-Man" Available:https://www.researchgate.net/publication/228577256_Teaching_Introductory_Artificial_Intelligence_with_Pac-Man

[3]. M. Barbehenn, Motorola GmbH, Munchen, Germany "A note on the complexity of Dijkstra's algorithm for graphs with weighted vertices"

[4]. Fausett, Laurene, "Fundamentals of Neural Networks Architectures, Algorithms, and Applications", Prentice-Hall,Inc, 1994.

[5]. Higgins, Dan., "Pathfinding Design Architecture", AI GameProgramming Wisdom, Charles River Media, 2002

[6]. Higgins, Dan., "GenericPathfinding", AI GameProgramming Wisdom, Charles River Media, 2002

[7]. Matthews, James, "Basic A* Pathfinding Made Simple", AI GameProgramming Wisdom, Charles River Media, 2002.

[8]. Russel, Stuart., Norvig,Peter., "ArtificialIntelligence A Modern Approach", Prentice-Hall,Inc, 1995

[9]. Stentz, Anthony., "Optimal and Efficient Path Planning for Partially-known Environments." In proceedings of the IEEE International Conference on Robotics and Automation, May 1994

[10]. Stentz, Anthony., "Map-Based Strategies for Robot Navigation in Unknown Environments". In proceedings of the AAAI Spring Symposium on Planning with Incomplete Information for Robot Problems, 1996

[11]. White, Stephen., Christensen, Christopher., "A Fast Approach to Navigation Meshes", Game Programming Gems 3, Charles River Media, 2002