IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 21, Issue 3, Ser. II (May - June 2019), PP 01-05
www.iosrjournals.org

Automated Gaming Using a Machine Learning Approach

Shraddha More[1], Rajat Mistry [2], Mayuresh Pachangane[3] , Amogh Kini[4]
Rishikesh Bhosale[5]

[1]*(St. John College of Engineering & Management, India)*
[2]*(St. John College of Engineering & Management, India)*
[3]*(St. John College of Engineering & Management, India)*
[4]*(St. John College of Engineering & Management, India)*
[5]*(St. John College of Engineering & Management, India)*
Corresponding Author: Shraddha More[1]

Abstract: *Platform games are being revived with both their 2D and 3D versions being developed. A major challenge faced by the industry is a lack of automated system-level approaches for game testing. Currently in most game development organizations, games are tested manually or using semi-automated techniques. Such testing techniques do not scale to the industry requirements where more systematic and repeatable approaches are required. We provide guidelines for modeling the platform games for testing using our testing model. We present the details related to automated test case generation, execution, and oracle generation. Machine Learning approach can help us analyze the current state and allows to take action on it.*

Keywords: *Automation, Gaming, Machine Learning, Deep Learning*

Date of Submission: 30-04-2019 Date of acceptance: 14-05-2019

I. Introduction

Artificial Intelligence (AI) plays an important role in modern computer games. The games can be improved to become more amusing through well designed AI, and players can also obtain better challenge experience from playing. Therefore, how to let machine have human-like thinking ability, is one of the most importance subjects in game AI nowadays. Machine learning methods in the area of artificial intelligence. The methodology is developed based on the concept of “trial-and-error”, and the result of each “trial-and-error” will be saved as a “delay reward”. The goal of Machine learning is to allow the machines have human-like thinking ability [1].

Machine learning (ML) is the study of various algorithms and statistics that the computer system uses in improving their performance on a specific task. ML algorithms uses a mathematical model, which is known as "training data", to make predictions or decisions without being explicitly programmed to perform the task. Machine learning is primarily related to computational statistics that focuses on making predictions using machines. The study of mathematical optimization delivers to the field of machine learning- methods, theory and application domains. Automation is a field of study within machine learning, and it focuses on exploratory data analysis through unsupervised learning. It is used across business problems, also it is referred to as predictive analytics [2].

Machine learning algorithms are further divided as supervised or unsupervised. Supervised machine learning algorithms can apply what has been learned earlier in the past to new data using labelled examples in order to predict future conditions. In contrast, unsupervised machine learning algorithms are used in case when the information used to train is neither classified nor it is labelled.

II. Literature Review

In 2014, Yung-Ping Fanget. al.[3] presented, Artificial Intelligence plays an important role in modern computer games, as a well-designed AI allows various games to become more entertaining and challenging. Therefore, how to give Non-Player-Characters (NPC's) more human-like thinking and working abilities and also give the players extra fun from the interaction with NPC's, are very important considerations in the area of game AI. Although there are so many types of games, the moving path of NPC's is usually a critical factor. If we want the NPC's to behave like human beings, the first thing is that they should move to the meeting point. Therefore, it is of interest to find the best moving track and policy for NPC's. In this case, the algorithm of reinforcement learning will be used as the main technique to solve the above problem. Reinforcement Learning is one of the unsupervised machine learning methods in the area of AI. The methodology has been developed that is based on the concept of “trial-and-error”, and the result of each “trial-and-error” action will be saved as a “delay reward”. In 2015, Michelle McPartland and Marcus Gallagher [4] provided the main disadvantage of reinforcement learning that is the algorithm needs to be run many times to get enough rewards for deciding suitable policy. Currently, some researchers have devoted themselves to dealing with this problem, such as by means of Hierarchical Reinforcement Learning (HRL). We will try to apply fuzzy logic in reinforcement learning to solve this problem. Fuzzy logic came into existence as a consequence of 1965’s fuzzy set theory by Zadeh, L. It has being applied to many fields of research, such as computer science, financial engineering, management, etc.

In 2015, Sidra Iftikhar Muhammad Zohaib Iqbal et.al[5] presented that, Game industry has recently emerged as a major software development industry. The number of games being developed has increased rapidly over the last few years. Major reasons for the increase in number of games include the availability of high processing display devices, powerful hand-held devices, mobile games and introduction of new game genres. Games are interactive, state-based, and are mostly real-time software that the end users can interact for a greater time period. A major problem faced by the gaming industry is the functional system-level testing automatically. A common approach of testing games is by manually playing a large number of potential scenarios that end users may exercise. A major focus of such testing is to ensure the correctness of games, for example, identifying the points where the game fails and checking for correct scoring. Similar to traditional software development, manual testing does not scale up to the requirements of game testing. Generating manual test case is a monotonous and error prone task, is not systematic and hence is not scalable. This becomes a major problem when changes appear frequently.

Reinforcement machine learning algorithm is a learning method that communicates with its own environment by producing actions and detects errors or rewards. Trial and error methods and provided reward state are the most relevant characteristics of this algorithm. Reinforcement learning thus allows machines and other software agents to automatically determine the ideal behavior inside a specific context to maximize its performance criteria. Simple reward feedback is required for the agent to learn which action is best and to be considered; this is hence called the reinforcement signal [1].

Deep Learning is a sub-field of machine learning many of the algorithms inspired by the structure and function of the brain called artificial neural networks (ANN). If you are just starting out in this field or you had some experience with neural networks some time ago, you may be a little confused. I know even I was confused initially and so were many of my colleagues and friends who learned and used neural networks in their projects. The leaders and experts in the field have ideas of what deep learning is and these specific and delicate perspectives puts up a lot of light on what deep learning actually is all about. In this past, you will discover exactly what deep learning is by hearing from a range of experts and leaders in the field.

III. Research Methodology

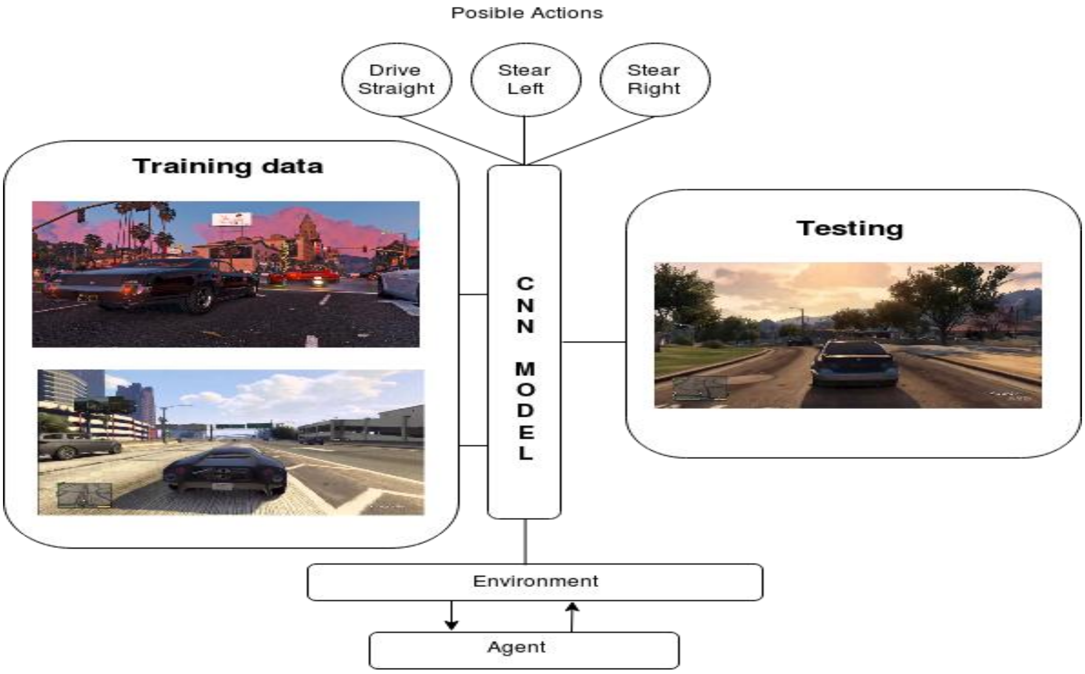


Fig 1: Proposed Architecture

Convolutional neural network. In machine learning, a network (CNN, or ConvNet) is a class of deep, feed-forward artificial neural networks, most commonly applied to analyzing visual imagery. In this system automation is carried out using ML approach. Training is done using the vehicles present in the game. Various conditions are to be considered regarding the environment present in the game. Here we have considered lanes of the road and also the surrounding objects present in the game. Training was done for a suitable amount of time and disk space. Finally testing is done in any of the parts of the game and the vehicle successfully drives on its own.

Though both supervised and reinforcement learning use mapping between input and output, unlike supervised learning where feedback provided to the agent is correct set of actions for performing a task, reinforcement learning uses rewards and punishment as signals for positive and negative behavior.

The working of the system is divided into four major phases. These are:
PHASE 1- The project is very useful for game developers [4]. The Gaming industry is exploding with a lot of crowd attracted towards gaming. For the most part, games are way behind the rest of the software engineering world when it comes to automated testing. People used to think that automated testing was impractical in something as complex as a video game [5], but as they have learned more about the process over the years, I've decided that may have been short-sighted. Thus, this has reduced the capital spend on Game Testers & allowed companies to build more games.

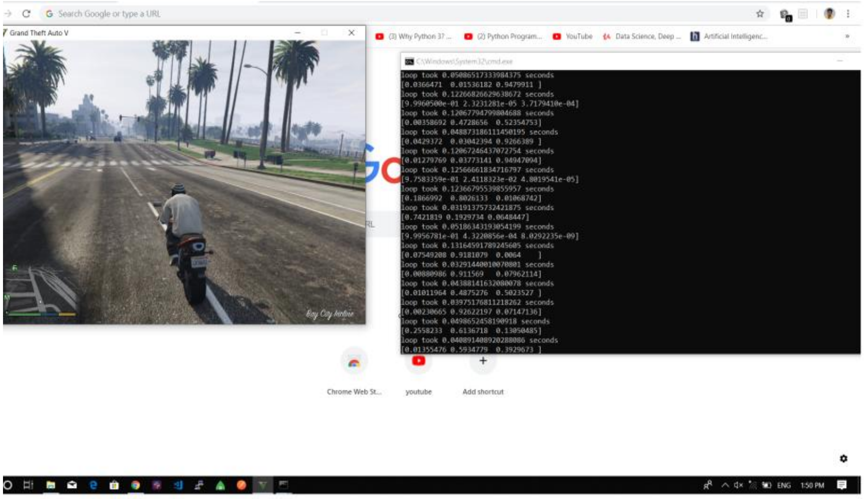
PHASE 2- We have tested our project on dinosaur game. The jump action taken by the dino is provided through pixel recognition. It jumps according to height of the obstacles and the distance between the obstacles and the dino. Thus, the dino can easily clear the obstacles[6]. As speed of game increases, the dinosaur can tackle the obstacles accordingly by judging the speed of the environment.

PHASE 3- Further we have made a automated car driving module using OpenCV. The car can move to different places through the lanes present on the road. This way the car driving and location modules can be tested. The side barriers and other vehicle obstacles are also taken into consideration. Using gray level processing and edge detection the car moves within the given lane.

PHASE 4 – Finally we made it fully automated; just the car driving part. The first thing we did is we trained the model using Tensorflow. Also, the testing phase needed to be given a lot of data and frames. We played the game numerous times for collecting a lot of data. The more we play the game and train the model, more is the accuracy. Thus, then we see the car driving on its own and take turn, explore the whole city.

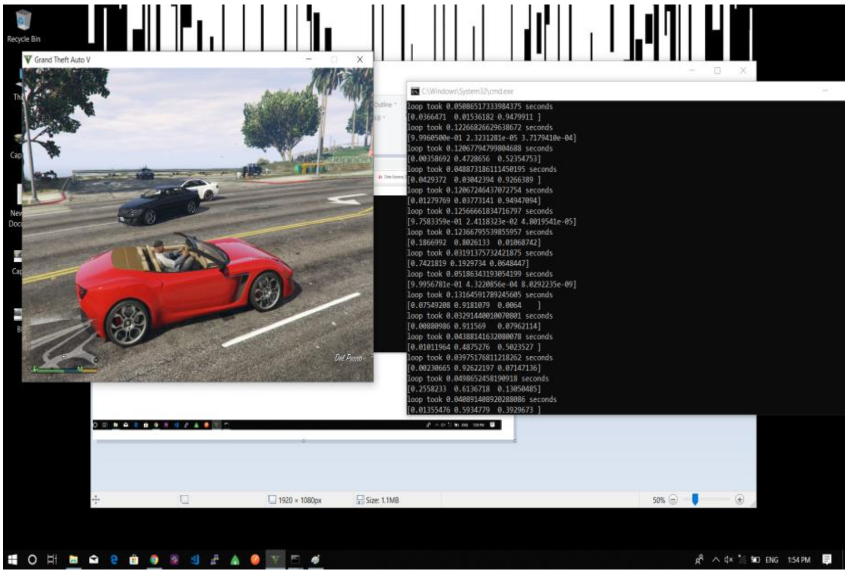
IV. Results And Discussions

In the first phase edge detection is performed using the Canny edge detection algorithm using open CV. This finds out the edges of the roads so, it can find out multiple edges around. If both the edges of both the sides intersect with each other then, the vehicle turns on the opposite side of the intersection. For example if both the edges intersect on the right hand side then the vehicle turns to the left and vice versa. In the second phase the model was trained according to the input through the keyboard. Each keyboard key was given a certain code and the actions to the keys were assigned accordingly. Further, we trained model to increase the accuracy rate. We used the actual gaming environment as the input for training our model. Using tensorflow supported GPU we trained our model. The train training and testing was carried out on high end GPU machine. The larger is the data the more time it requires for training. We collected a data of around 300 entries and estimated time was around 15-16 hours for basic level training and testing.



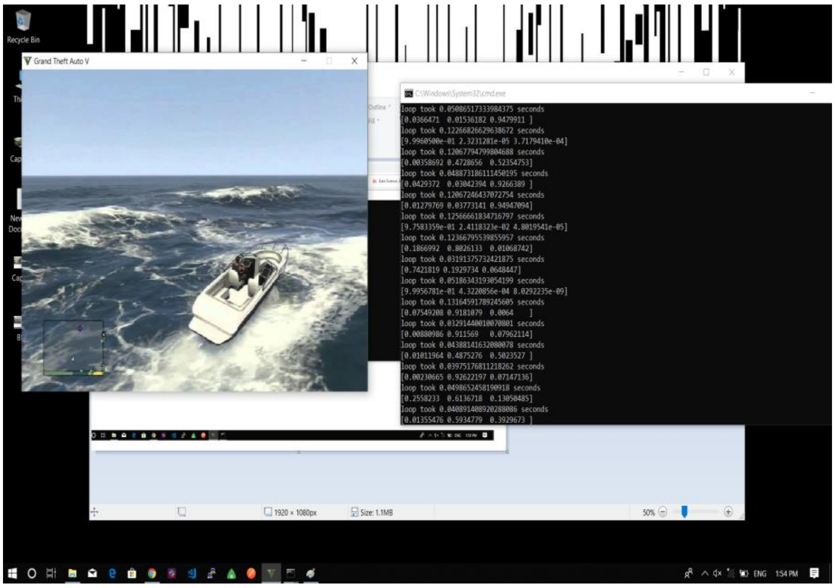
```
loop took 0.05086517333984375 seconds
[0.0366471  0.01536182 0.9479911 ]
loop took 0.12266826629638672 seconds
[9.9960500e-01 2.3231281e-05 3.7179410e-04]
loop took 0.12067794799804688 seconds
[0.00358692 0.4728656  0.52354753]
loop took 0.048873186111450195 seconds
[0.0429372  0.03042394 0.9266389 ]
loop took 0.12067246437072754 seconds
[0.01279769 0.03773141 0.94947094]
loop took 0.12566661834716797 seconds
[9.7583359e-01 2.4118323e-02 4.8019541e-05]
loop took 0.12366795539855957 seconds
[0.1866992  0.8026133  0.01068742]
loop took 0.03191375732421875 seconds
[0.7421819 0.1929734 0.0648447]
loop took 0.05186343193054199 seconds
[9.9956781e-01 4.3220856e-04 8.0292235e-09]
loop took 0.13164591789245605 seconds
[0.07549208 0.9181079  0.0064    ]
loop took 0.03291440010070801 seconds
[0.00880986 0.911569   0.07962114]
loop took 0.04388141632080078 seconds
[0.01011964 0.4875276  0.5023527 ]
loop took 0.03975176811218262 seconds
[0.00230665 0.92622197 0.07147136]
loop took 0.04986524581909918 seconds
[0.2558233  0.6136718  0.13050485]
loop took 0.04089140892028808 seconds
[0.01355476 0.5934779  0.3929673 ]
```

Fig 2: Stage 1: Edges Detected



```
loop took 0.05086517333984375 seconds
[0.0366471  0.01536182 0.9479911 ]
loop took 0.12266826629638672 seconds
[9.9960500e-01 2.3231281e-05 3.7179410e-04]
loop took 0.12067794799804688 seconds
[0.00358692 0.4728656  0.52354753]
loop took 0.048873186111450195 seconds
[0.0429372  0.03042394 0.9266389 ]
loop took 0.12067246437072754 seconds
[0.01279769 0.03773141 0.94947094]
loop took 0.12566661834716797 seconds
[9.7583359e-01 2.4118323e-02 4.8019541e-05]
loop took 0.12366795539855957 seconds
[0.1866992  0.8026133  0.01068742]
loop took 0.03191375732421875 seconds
[0.7421819 0.1929734 0.0648447]
loop took 0.05186343193054199 seconds
[9.9956781e-01 4.3220856e-04 8.0292235e-09]
loop took 0.13164591789245605 seconds
[0.07549208 0.9181079  0.0064    ]
loop took 0.03291440010070801 seconds
[0.00880986 0.911569   0.07962114]
loop took 0.04388141632080078 seconds
[0.01011964 0.4875276  0.5023527 ]
loop took 0.03975176811218262 seconds
[0.00230665 0.92622197 0.07147136]
loop took 0.04986524581909918 seconds
[0.2558233  0.6136718  0.13050485]
loop took 0.04089140892028808 seconds
[0.01355476 0.5934779  0.3929673 ]
```

Fig 3: Stage 2: Working model with least inputs



```
loop took 0.05086517333984375 seconds
[0.0366471  0.01536182 0.9479911 ]
loop took 0.12266826629638672 seconds
[9.9960500e-01 2.3231281e-05 3.7179410e-04]
loop took 0.12067794799804688 seconds
[0.00358692 0.4728656  0.52354753]
loop took 0.048873186111450195 seconds
[0.0429372  0.03042394 0.9266389 ]
loop took 0.12067246437072754 seconds
[0.01279769 0.03773141 0.94947094]
loop took 0.12566661834716797 seconds
[9.7583359e-01 2.4118323e-02 4.8019541e-05]
loop took 0.12366795539855957 seconds
[0.1866992  0.8026133  0.01068742]
loop took 0.03191375732421875 seconds
[0.7421819 0.1929734 0.0648447]
loop took 0.05186343193054199 seconds
[9.9956781e-01 4.3220856e-04 8.0292235e-09]
loop took 0.13164591789245605 seconds
[0.07549208 0.9181079  0.0064    ]
loop took 0.03291440010070801 seconds
[0.00880986 0.911569   0.07962114]
loop took 0.04388141632080078 seconds
[0.01011964 0.4875276  0.5023527 ]
loop took 0.03975176811218262 seconds
[0.00230665 0.92622197 0.07147136]
loop took 0.04986524581909918 seconds
[0.2558233  0.6136718  0.13050485]
loop took 0.04089140892028808 seconds
[0.01355476 0.5934779  0.3929673 ]
```

Fig 4: Stage 3: Different vehicles tested

V. Conclusion And Future Scope

Testing anything can be a very hectic as well as very time consuming for the person or laborer as well as the company or group testing it. So, considering games testing it is not very scalable in Industries. So, automating a game testing can not only have better performance but also help with the time of testing. So, methodology of machine learning can provide AI agents with a good ability to act as a human being. There are still many shortcomings with machine learning is that it needs a large amount of data and a lot time to get trained with that data and also test the game with it. Our experience shows that creating test ready models is better that manual testing in all games.

So, in near future we are expecting to find a better way out of this to the problem. And also, ways to work it around with the multi-agents, make a perfect AI without major mistakes and try to be a more efficient one in area of game testing. Also, the game testing can get fully automated as in a click of key. The complexity of automation increase can give out better results.

References

[1] Yung-Ping Fang, “Applying reinforcement learning for game AI in tank battle game” ,Fourth International Conference on Innovative Computing, Information and Control, 2009

[2] Michelle McPartland and Marcus Gallagher, “Reinforcement learning in first person shooter game” Member, IEEE transactions on computational intelligence and AI in games, vol. 3, NO.1,March 2011 .

[3] Muhammad Zohaib Iqbal Sidra Iftikhar Wardah Mahmood Muhammad Uzair Khan, “An Automated Model Based Testing Approach for Platform Games”, 2015 IEEE.

[4] Sendex , “Self driving car neural network in the city - Python plays GTA with Tensor Flow” , Published on Apr 21, 2017, [Video file]. Available: https://youtu.be/KSX2psajYrg,

[5] Siraj Raval, “How to Simulate a Self-Driving Car”, Published on May 17, 2017 , , [Video file]. Available: https://youtu.be/EaY5QiZwSP4

[6] The PC Geek, Making Google's Dinosaur Bot (PYTHON) ,Published on Sep 27, 2017, [Video file]. Available: https://youtu.be/bf_UOFFaHiY

[7] Learn DL Code TF, Understanding AlexNet architecture, Published on Oct 30, 2018, [Video file]. Available: https://youtu.be/uLddd86qVFs

[8] Shraddha More, Arpit Rohela, “Vulnerability Assessment and Penetration Testing through Artificial Intelligence” in International Journal of Recent Trends in Engineering & Research, DOI:10.23883/IJRTER.2018.4028.RDD10, ISSN: 2455–1457 Vol.4, Issue 1, pp.217-224, Jan 2018.

[9] Shraddha More, Anagha Mudrale, Sukhada Raut, “Secure Transaction System using Collective Approach of Steganography and Visual Cryptography” in IEEE 2018 International Conference on Smart City and Emerging Technologies (ICSCET 2018), 5th Jan 2018.

Shraddha More[1]" Automated Gaming Using A Machine Learning Approach" IOSR Journal of Computer Engineering (IOSR-JCE) 21.3 (2019): 01-05.