The Bot - God of Tetris

Sachin Kumar Sahu
Amity University Greater Noida
sachin_vns101@yahoo.com

Abstract **-** This research is an implementation of Artificial Intelligence in a game called Tetris which is done by the use of Reinforcement Learning Specifically Genetic Algorithm. This research provides an overview of the future related developments related to artificial intelligence with the integrated use of genetic algorithm used for wide use or commercial use coming into the real picture. This algorithm focuses developing itself or furthering itself towards the point of perfection. It can also pave the way for revolutionizing existing technology and creating efficient machines through the implementation of genetic algorithms and the data collected by them.
At the very start of this research the game bot doesn't know anything or had any specific coding for that purpose but with the help of reinforcement learning, the game bot not only learned but also perfected itself till the point it became the only best version existing for that game. One question that intrigues, even more, is that, is there any possibility of something even better than this? The interesting fact about this development is that though it took a considerable amount of time for the AI to become the best existing version, it becomes very evident that with the use of even more simplified and efficient ways this development can go even further and it can also be incorporated into other games, machines with just some slight modifications.
In this research, I have used an unending game called Tetris and the way the bot plays this game shows the credibility of reinforcement learning which progresses with every step.

Keywords **–** *Tetris, video game automation, evolution, reproduction, Artificial Intelligence, bot, evolutionary algorithm*

I. Introduction

Machine learning is a field of artificial intelligence which deals with an algorithm that solves problems (usually used in complex problems) based on data provided and using this data the algorithm or machine creates empirical information which is also called experience.

Games have had a long history of being a testbed for AI ever since the days of Pong (Brick game of the 90s). Traditionally, game programmers have taken a 'Reductionist Approach' to building an AI. They've reduced the simulated world to a model and had the AI act on prior knowledge of that model that did not work out for the most part. All the world models are different so we couldn't feed it just one world model. Instead of modeling the world, we need to model the mind. We want to create an AI that can become a pro at any game we throw at it. So, in thinking about this problem, we have to ask ourselves what is the efficient way to do this? The answer is Artificial General Intelligence (AGI), that's one algorithm that can solve any problem with human-level thinking or greater.

One of the tortuous problems in the real world is solving a never-ending game where the bot will never know the reward threshold at which the game will be considered as solved. One such game is Tetris which is

both unending and popular. This paper presents Genetic Algorithm also known as an Evolutionary algorithm to create a bot which can play this game infinitely without having the complexities involved in other algorithms in reinforcement learning.

II. AI and Game – Eternal Relation

Video games have been around since the 50's when Joseph Kates publicly demoed Tic-Tac-Toe at the Canadian National Exposition. That bot used simple scripted actions that ran the same way every time regardless of whatever move the player made. His demo got people hyped, though, because no one had ever seen a computer play a game before and they were lining up off the block to check it out. The game bots that were invented afterward for games like Nim and Space-war were similar but along came Pong. The Pong bot's paddle had to make decisions based on the human player's actions, and that made it feel more realistic. Pong marked the beginning of using heuristics to create game bots. Heuristics are educated guesses and pretty much every single video game bot since Pong's has used them. A bot will map out a possible set of decisions as a tree of possibilities, then use one of many techniques to pick the best one. But it's still always boiled down to a bunch of if-then statements.

If Pac-Man moves this way, then the blue ghost should move this way, If Master Chief sees a Grunt, then it should move in circles. But yes, video game bots have pretty much always boring and sucked because there are only so many edge cases that a programmer can predict.

Therefore, we need to think about this problem differently. When people start playing a game, they don't know anything

about its environment beforehand. The hallmark of intelligence is our ability to generalize, but in order to create the artificial intelligence that can generalize to solve any task, there has recently taken pace.

Recently, a team of researchers at DeepMind now owned Google got close to creating one bot that could beat almost any Atari game knowing literally nothing about the game beforehand. No game-specific hard-coded rules at all. It took the raw pixels of the game as input and its controls. Using those two things, it learned how to beat almost any Atari game it was given.

Machine learning is a subset of artificial intelligence where computer algorithms are used to independently learn from data and information. In computers learning automation does not have to be clearly programmed but can change and improve their own algorithms in their own right.

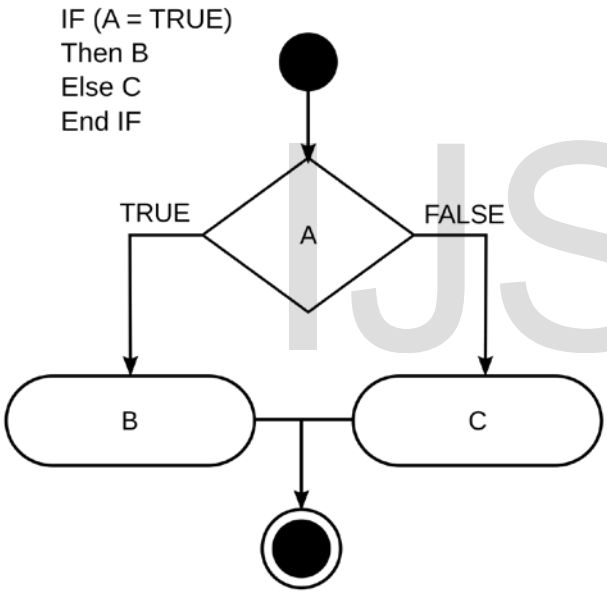

Fig. 1. Traditional if and else data flow diagram.

If you take a Machine learning algorithm, and feed it lots of data and compute, it can learn to do a whole lot of incredible things. The field of deep learning right now is where physics was in the early 1900's.

III. Implementation & Complexities

We created the game "Tetris" under its natural environment. The game had full controls but instead of us playing the game we decided to create an AI to play the game for us i.e. self-playing player also known as Game Bot and to create BOT we've successfully used Reinforcement learning-based algorithms.

- *Rules* –

1. The main board of Tetris is of size 10x20.

2. There are seven types of blocks that can fall i.e. "I", "O", "J", "L", "S", "Z", "T".

3. Each time a block moves down the score is incremented by one and when a row is cleared row disappears and then we get we gain a certain amount right to the score.

4. The goal is to get to 500 (for faster mutation) without the block stacking up over the ceiling.

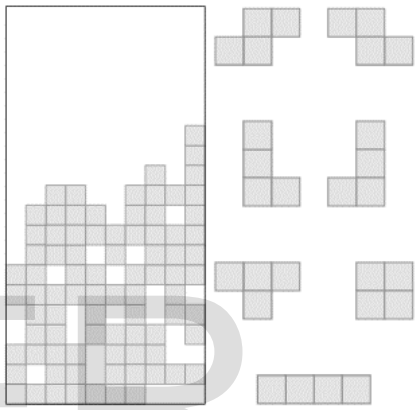

Fig. 2. Pictorial representation of Tetris board and blocks

- ***Objectives*** - Our objective is to eliminate as many lines as possible to make the environment height as little as possible in order make as many moves as possible to reach a high score.

  To achieve this objective, our BOT will determine the best move for a particular block of Tetris that brute-force all possible moves (rotation and position) and Calculate a score for each possible move (using the search track) and select it with the highest score as the next move.

- ***Working implementing Genetic Algorithm to the game -***
  1. We initialize a population of 50 genomes.
  2. Each genome is a set of seven parameters of the game state
  3. Our AI makes moves based on these parameters value
  4. It tries out all possible moves based on each genome in the current population to make a single move
  5. Since Tetris is a never-ending game, therefore, we will set the goal is to get a score of at least 500 for fast mutation. We keep evolving until then.

IV. Making of God

When all is said in done, a diversion playing specialist chooses which activity to pick next at any given condition of the amusement in view of the reward it will increase subsequent to playing out that activity. The specialist picks, among a few conceivable activities, the one that will restore the best result. It is however troublesome and at some point, impractical to judge the estimation of an activity, and one route around this is to judge the estimation of the express the activity will prompt. By and large, this is accomplished by utilizing an assessment work that doles out numerically esteems to amusement states. The player hence picks the activity prompting the best-esteemed state. In the game like Tetris, the placement of blocks depends very much on the certain condition. Also known as rewarding and penalty factor. Through which the AI will decide the best possible move but at the starting, we don't know the optimum value of this factor and therefore in order to know them, we'll be using the Genetic algorithm also known as Evolutionary Algorithm.

Genetic Algorithms inculcates the survival of the fittest in such a way that generations are maintained, and from one generation, only those characters pass the next generation which has survived/performed well in the previous generation. Each generation is formulated by a set of strings, which are in the sink to the chromosomes in the DNA. Cross-Over, Mutations and other biological effects occur which affect the offspring in the next generation. The same is true for the genetic algorithm too. Each individual represents a point in a search space and all kind of possible solution. The individuals present in the population are then made to go through a process of evolution.

In biology, individuals compete for resources, some of them win and others lose. The winning individuals are very much likely to contribute more share to the offspring in comparison to the losing ones. Offspring that inherit features from two winning individuals has a good chance of being better than each of its parent. The logic remains the same her also. Some of the important terms similar to Genetic Algorithms are explained below:

1. ***Search Space*** - A collection of individuals is maintained in a search space. Each individual represents a solution to a problem (in our case, this is weights of the edges in the Neural Net). Each and every individual is coded as a string representing it. Individuals are like chromosomes, and the entries in that vector representing an individual are like genes. The fitness function defines the competing ability of an individual. The aim is to make the offspring produced by this generation inherit the best properties of any two individuals from this generation.
2. ***Selection*** - In the biological analogy, the individual that is "fittest" is passed on to the next generation. In our implementation too, the individuals with the top 5 fitness values pass it on to the next generation without any hindrance.
3. ***CrossOver -*** It is a condition in which two individuals are selected from a generation and a position in their bit strings is chosen at random. The bits corresponding to this position are swapped between the too. Now, this offspring also passes on to the next generation. As this combines good individuals too, it has very slight chance of showing up an individual which is better than the current generation.
4. ***Mutation*** - In here, the bits of some people are flipped at some randomly chosen position. This is to ensure that the solution doesn't converge to an answer which is not universally optimal and that there is always a chance to revert back, and not converge to a wrong point.

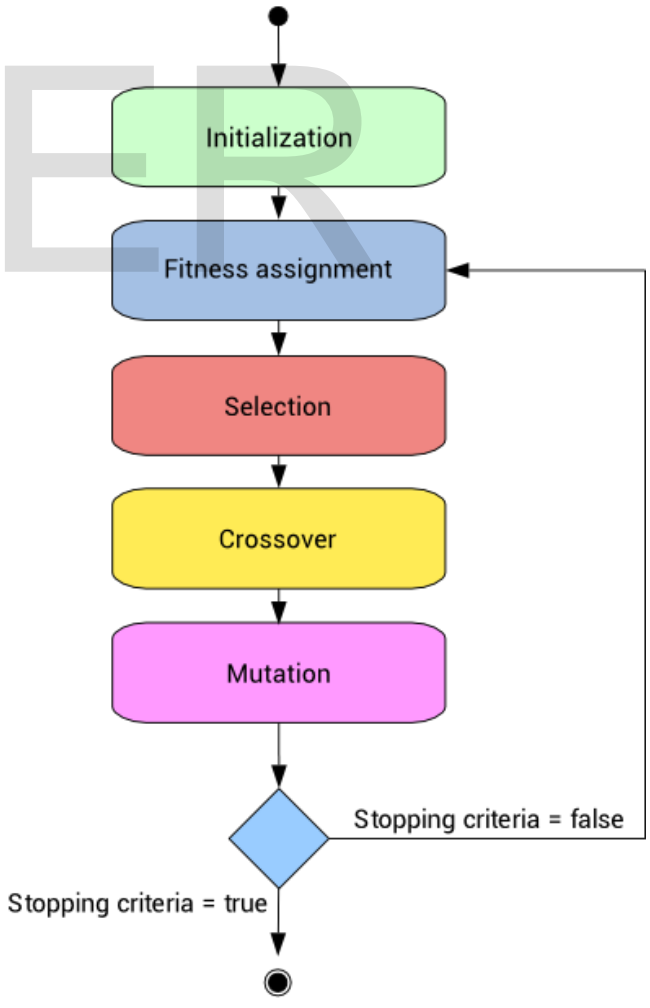

Fig. 3. Genetic Algorithm Sequence

V. Performance Check

Fig. 4, can be understood by knowing the parameters mentioned, depicts the first-generation performance.

Value	Description
id	Randomly generated unique identifier for a genome
rowsCleared	The weight of each row cleared by the given move. the more rows that are cleared, the more this weight increases
weightedHeight	The absolute height of the highest column to the power of 1.5, added so that the algorithm can be able to detect if the blocks are stacking too high
cumulativeHeight	The sum of all the column's heights
relativeHeight	The highest column minus the lowest column
holes	The sum of all the empty cells that have a block above them (basically, cells that are unable to be filled)
roughness	The sum of absolute differences between the height of each column, (for example, if all the shapes on the grid lie completely flat, then the roughness would equal 0)

To understand the table, the value should be justified as the numbers which are in positive are considered as a reward and are important for the bot to finish the game and the number which is negative are considered to be as penalty and should be avoided.


```
{
  "id": 0.521484701164314
  "rowsCleared": -0.24566207610548485
  "weightedHeight": 0.4638535466003274
  "cumulativeHeight": -0.01775524301123943
  "relativeHeight": 0.3461601104679859
  "holes": 0.042788205355442144
  "roughness": 0.20300688172044878
}
```


Fig. 4. Generation 1 fitness score evaluated by bot

There is a noticeable difference between the two generations which shows that it took more than 25 generations for the bot to improvise to the point of perfection. When we consider the amount of time then the 25th generation agent was able to play a bit more than 240 hours after which it was carried by the next generation and still in action even after more than 1000 hours.


```
{
  "id": 0.5018265243129169
  "rowsCleared": 0.4106816860123958
  "weightedHeight": -0.08679520494876472
  "cumulativeHeight": -0.6152727732730796
  "relativeHeight": 0.028340097577794654
  "holes": -0.17437734216356443
  "roughness": -0.17720415654757266
  "fitness": -1
}
```


Fig. 5. Generation 25 fitness score evaluated by bot

Talking on a technical note then the bot started initially with the average value as mentioned by us which worked as a reference for the bot to start developing its way through the problem. After, several brute forcing attempts the bot came to the real picture of performance with an impressive prolonged duration of play which was faster and didn't want to give up even after 1000 hours, it can be assumed that barring it stops, it would be perfect or else it would take another 2 or 3 generations to reach the optimum performance which can be considered ideal and all of this thing has been done without the support of human being.

VI. Results

The bot at the initial stage as fragile as a human baby and thus we had to provide it with a point to begin. unlike a human baby, the bot had the capability to learn faster and go through evolutions with even greater speed.

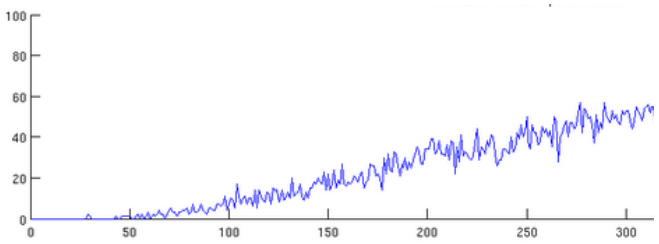

Fig. 6. Time wise progression and improvement

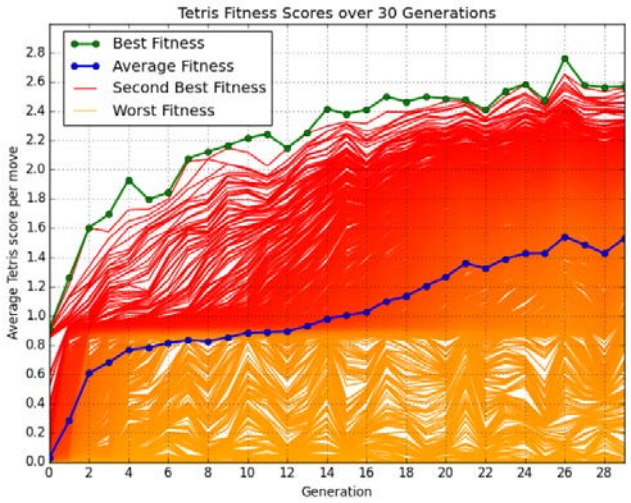

Fig. 7. Fitness score improvement on each and every move and its impact on the generation

VII. FUTURE

We are living in this world which is fully digitized and will be extremely digitized in near future stranding towards ultra-digitized in far future but keeping in mind the aspects of reinforced learning and its implication to the fields of artificial intelligent, it can be seen as a possible future where machines will take over the most of the works done by human and will do it with sheer perfection. Another issue that comes to mind is the concept of AI taking over the human race which is yet a hypothetical assumption and should rather be restricted to that only because an implication of such an algorithm "Genetic Algorithm" can help the machine to learn and evolve very fast, way faster than our biological limits.

Talking more specific than it becomes imperative to mention the economic changes that will be coming to the picture in the near distant future. For example, in the near future most of the work will be done by machine and therefore most human being will be unemployed and hence the requirement for basic salary for everyone. So, the genetic algorithm can be used in this area to find the perfect match of salary with respect to market and machines.

VIII. CONCLUSION

As we do come at the very end of this report, it's our duty to let you know that this work has only been possible due to the modern and ever upgrading technology which really led us close to the objective of benefitting the present modern scenario with the help of wide applications of Artificial Intelligent which really helped us reach the boundaries to know and to fathom the horizons of Artificial Intelligent.

Not everything can be understandable at first, real world is complicated and messy even for full-grown human beings so in order to go further towards the success of creating Artificial General Intelligent we'll have to train them on the closet thing we have like real world; the thing which requires the logic and then we'll have to wait for machine to solve that problem logically only then the machine will be able to learn and grow.

As we conclude we see it as a situation where in we need to tell the world that computer science is highly efficient to pave the way for a modern brain that works as per our wants.

We hope that this report would have been very practical while marking out the direction of our work.

IX. ACKNOWLEDGMENT

Going through this research it was so fun a task to come across things we actually didn't know and didn't imagine about the complexity and sophistication of the modern-day codes that are capable of having their own brains and the top fact that a brain created such thing. We didn't know the vastness of this topic, it all sounded like a fiction and when this fiction actually became a reality for us we got know that it already began years ago. Our research started with the fascination related to Artificial intelligent but at the end, this thing led us to know more and more and our approach focuses on the practical aspects used these days and the coming scenario that would be faced and a practical solution to it. So, it can be said that this research began with a big blow to our knowledge which ignited our minds to analyze and to make this research reach everyone's mind so that people can see the future unfolding.

REFERENCES

[1] Demaine, E.D., Hohenberger, S., Liben-Nowell, D.: Tetris is hard, even to approximate. In: Warnow, T.J., Zhu, B. (eds.) COCOON 2003. LNCS, vol. 2697, pp. 351–363. Springer, Heidelberg (2003)

[2] Goldberg, D.: Genetic algorithms in search, optimization, and machine learning. Addoson-Wesley (1989)

[3] Perl, J.: Heuristics: Intelligence Search Strategies for Computer Problem Solving. Addoson-Wesley (1984)

[4] Boumaza, A.: On the evolution of artificial Tetris players. In: Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG 2009), pp. 387–393 (2009)

[5] Carr, D.: Adapting reinforcement learning to Tetris BSc (Honours) Thesis, Rhodes University (2005)

[6] Demaine, E.D., Hohenberger, S., Liben-Nowell, D.: Tetris is hard, even to approximate. In: Warnow, T.J., Zhu, B. (eds.) COCOON 2003. LNCS, vol. 2697, pp. 351–363. Springer, Heidelberg (2003)

[7] Langenhoven, L., van Heerden, W.S., Engelbrecht, A.P.: Swarm Tetris: Applying particle swarm optimisation to Tetris. In: Proceedings of

the IEEE Congress on Evolutionary Computation (CEC 2010), Barcelona, pp. 1–8 (2010)

[8] The psychology of randomness by Shawn Hargreaves (2009) > https://blogs.msdn.microsoft.com/shawnhar/2009/12/17/the-psychology-of-randomness/

X. Author Information

Sachin Kumar Sahu, Student, Bachelor of Computer Science & Technology – 3C (2015-19) at Amity University, Greater Noida.