Implementing Reinforcement Learning on MENACE

A dual degree project submitted by

ALEKHYA BHATRAJU (EE15B081)

In partial fulfilment of requirements for the award of the dual degree of

BACHELOR OF TECHNOLOGY IN

ELECTRICAL ENGINEERING

AND

MASTER OF TECHNOLOGY IN

ELECTRICAL ENGINEERING

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

MAY 2020

CERTIFICATE

This is to certify that the project titled ***"Implementing Reinforcement Learning on MENACE"***, submitted by Miss Alekhya Bhatraju(EE15B081), to the Indian Institute of Technology Madras, for the award of the degrees of Bachelor of Technology in Electrical Engineering and Master of Technology in Electrical Engineering, is a *bona fide* record of the research work carried out by her under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Bharath Bhikkaji
Project Advisor
Professor
Department of Electrical Engineering
Indian Institute of Technology Madras

Dr. David Kolli Pillai
Head of the Department
Professor
Department of Electrical Engineering

Place: Chennai

Date: July 6, 2020

ACKNOWLEDGEMENTS

First and Foremost, I am very grateful to the Prof.Bharath Bhikkaji, Department of Electrical Engineering, Indian Institute of Technology Madras, for guiding me throughout the project.

His support during the course of this project is really admirable. I really am grateful for his patience in explaining the unknown concepts to me regardless of the time.

I would also like to thank my friends for their time, and also for helping me to brainstorm the ideas. The discussions I had with them really helped me to improve myself.

Not least of all, I would like to thank my family for their unrelenting emotional support. Their emotional support played a huge role in completion of this project.

Alekhya Bhatraju.

ABSTRACT

In this thesis, we learn about machine learning through a game called Noughts and Crosses (tic-tac-toe).

Donald Michie has invented a machine called MENACE (Machine Educable Noughts and Crosses Engine) during 1961.

In this thesis we try to understand how Michie has made just a pile of matchboxes to learn. And we will also learn about the learning ability of MENACE and how it differs from a human learning this game for the first time. We try to implement new methods of Machine learning to improve the efficiency of MENACE.

We come across Q-learning method (one of the learning methods in Reinforcement learning), which could be applied in the noughts and crosses game. By varying the parameters, we finally achieve a process with better efficiency for the MENACE.

LIST OF CONTENTS

CHAPTER 1

1.1 Introduction:

Machine Learning is a widely used method in the world right now to deal with the challenges we face in daily life. We encounter uses of Machine learning in several ways like recommendation systems, language translation systems, speech recognition systems etc.

As we all know machine learning is about a machine, learning to do some tasks(T) given to it by itself, using its experience(E). In this process of learning, machine tries to improve its performance(P) on tasks (T), using its experience(E), automatically without the need of being programmed by humans at every step.

Machine learning can be broadly divided into 3 types:
1) Supervised Learning
2) Unsupervised Learning
3) Reinforcement Learning (explained in later chapters)

Now to get a better understanding on machine learning, let's trace our history back to the origin of machine learning. Researchers have used methods similar to machine learning even before the word "Machine Learning" is coined. The journey of machine learning began simply with games, by making the machine learn through playing games.

NOUGHTS AND CROSSES GAME:

In this thesis, we will concentrate on a machine called "MENACE". MENACE is created by Donald Michie in 1961. Donald Michie was a researcher from England, he had done commendable works in subjects related to Artificial Intelligence and Game Theory.

MENACE stands for Machine Educable Noughts and Crosses Engine. As the name clearly suggests it is about a machine, learning to play Noughts and Crosses game against human.

Now let's briefly learn about how MENACE works before going into deep analysis about it. MENACE is a machine which plays tic-tac-toe game against human and learns by itself to not lose in the game.

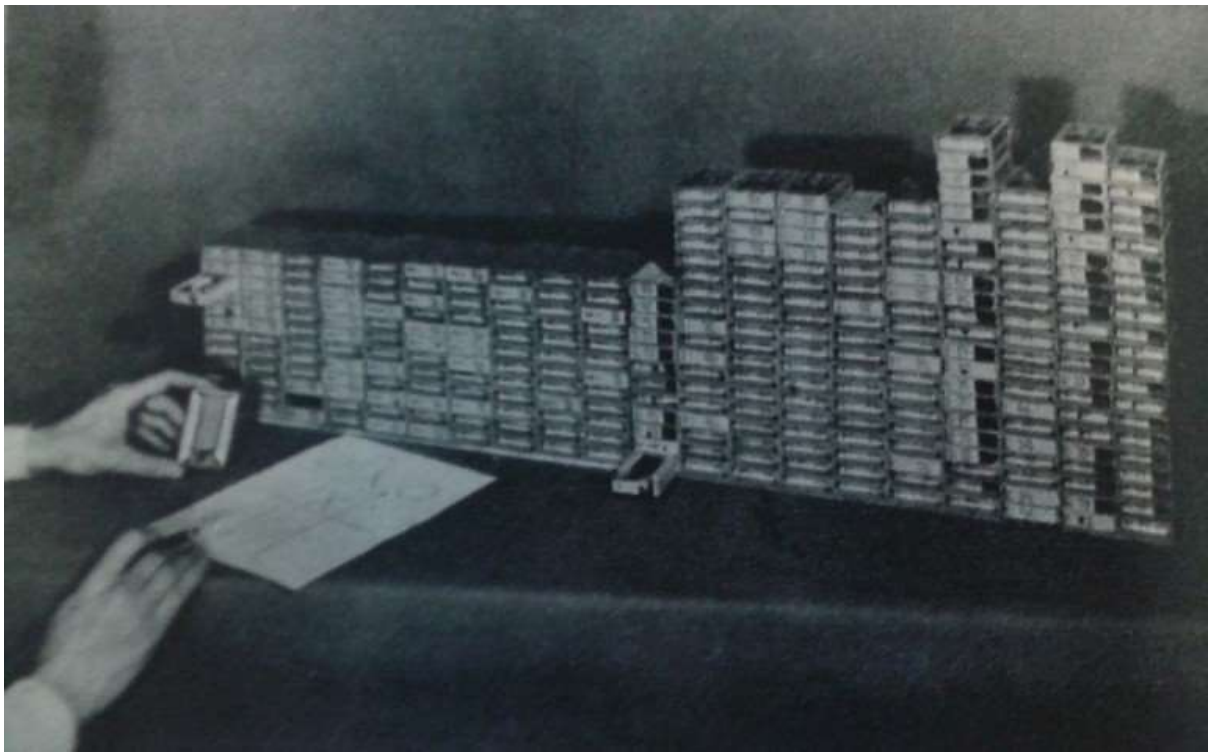


FIGURE 1:DONALD MICHIE PLAYING TIC-TAC-TOE GAME AGAINST PILE OF MATCHBOXES

MENACE is actually built from a collection of matchboxes (304 accurately). As you can observe from the picture, matchboxes are all put together using an adhesive, so that they can stay intact throughout the game. Each of these matchboxes had a certain configuration drawn on them. These configurations correspond to the state of the game that MENACE is currently in. Each matchbox also has beads placed inside them. These beads are in nine different colours. Each colour corresponds to different positions on the board (3x3 table, where game is being played).

Let's see an example of how the game generally works. Since Donald Michie always made MENACE play the first move, we would also consider for this example that MENACE makes first move. First we have an empty board with nine empty squares. Let's assume MENACE makes its first move at the position 5, that is at the centre of the board with 'O'. Now, Michie plays 'X' at any random position (here 2nd position). MENACE replies with 'O' at the position 8 (this position is decided by the colour of bead drawn), and so on the game continues. Michie wins the game by his 4th move, MENACE fails to defend against human on its first try.

	O	

	X	
	O	

	X	
	O	
	O	

	X	X
	O	
	O	

	X	X
	O	O
	O	

	X	X
X	O	O
	O	

	X	X
X	O	O
O	O	

X	X	X
X	O	O
O	O	

TABLE 1:BOARD CONFIGURATIONS DRAWN ON MATCHBOXES

MENACE's moves are decided at each move by observing the current configuration of game which is being kept under track by Donald Michie. And then picking a random coloured bead from drawer of the match box which matches the said configuration. Based on the colour of the bead MENACE's position is decided.

The colours of the beads according to the original paper are as given:

White	Lilac	Silver
Black	Gold	Green
Amber	Red	Pink

TABLE 2:NINE DIFFERENT COLOURS OF BEADS REPRESENTING NINE DIFFERENT POSITIONS ON BOARD

So Michie observes the colour from that particular matchbox and makes the move for the MENACE. One important thing to note is after a bead is removed, it is placed on the table in front of the matchbox drawn.

How many beads does each matchbox contain?

Each matchbox has number of beads corresponding to possible number of moves in the present state of the game. So if we consider for MENACE's first move we have 36 beads (9 different colour beads x 4 beads for each possible move). Similarly, for MENACE's second move (third move overall) we have 21 beads. For MENACE's third move (fifth move overall) we have 10 beads. For MENACE's last move we have 3 beads for 3 possibilities.

Donald Michie continues to play games against 'MENACE' until 'MENACE' learns to either win or draw.

How did Michie make MENACE learn?

We all know that moves made by MENACE are completely random. So let's us consider a situation where MENACE lost to Michie (human). Then Michie removes the bead that was already drawn out from the match box during the game. We call this as Negative reinforcement. We are punishing the machine since it lost to human while playing that particular move. This way, the probability of machine playing that particular bead in its fourth move next time decreases. It avoids the move which could result in its loss.

Now, if Machine draws with the human, then we reward it by adding one more bead of same colour along with the bead already drawn out. This is Positive reinforcement. Since that particular move earned machine a draw in the game, adding beads increases the probability of machine not losing.

Similarly, if machine wins against human we reward it with additional 3 beads of the same colour along with the bead already drawn out. This is also Positive

reinforcement. Since that particular move earned machine a win in the game, adding 3 additional beads increases the probability of it winning more.

But since MENACE is playing against a human (Michie), it may not win even if takes certain precautions. But by following the above rules, and also by playing a lot of games, it will adapt to not lose. If MENACE plays this game against human for around 200 times, you will see that MENACE rarely loses as time pass by.

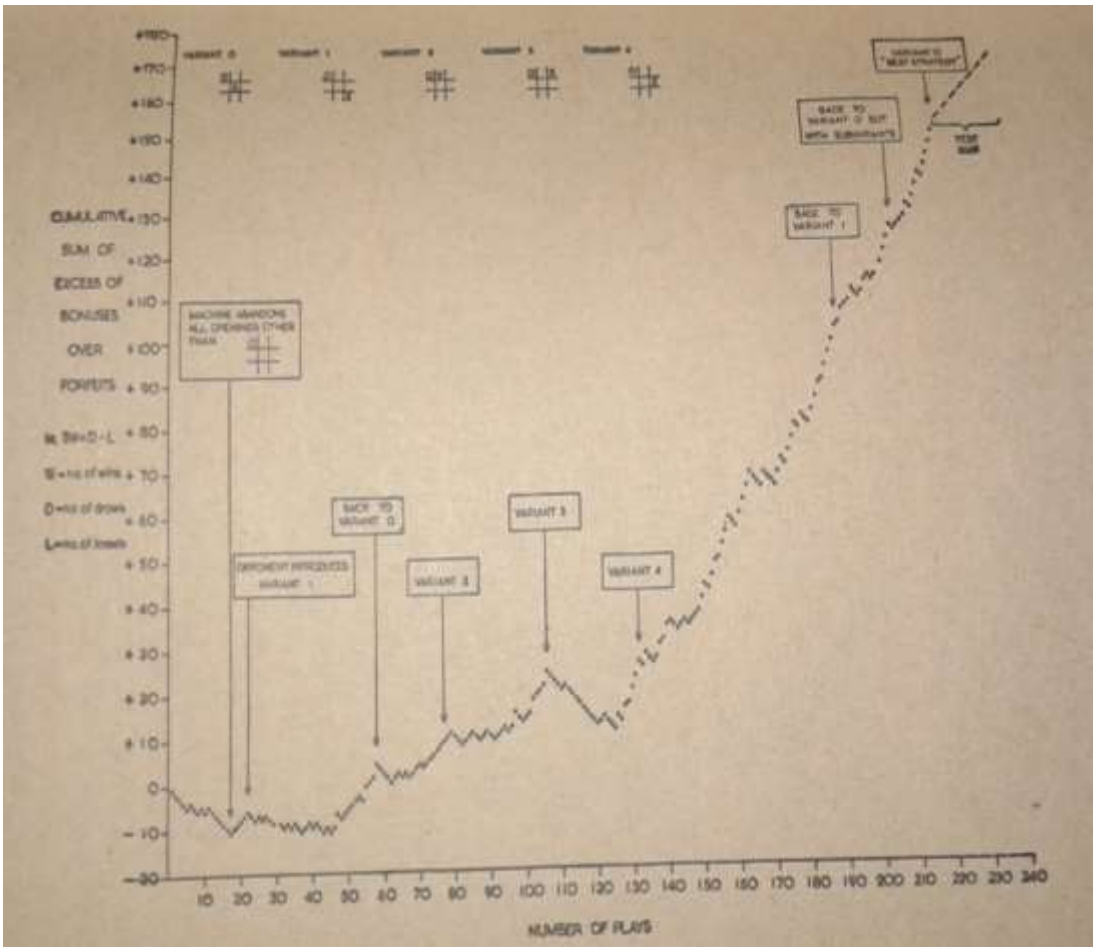


FIGURE 2:ORIGINAL GRAPH OBTAINED BY DONALD MICHIE

This is the original graph corresponding to the game tournament done by Donald Michie. The graph falls down by one unit when MENACE loses and rises by one unit when there is a draw and rises by 3 units when MENACE wins. You can observe that there is barely any drop in the graph as the number of games increases. Implies that the MENACE is slowly learning to not lose anymore.

The slope of the graph shows the expertise of MENACE.

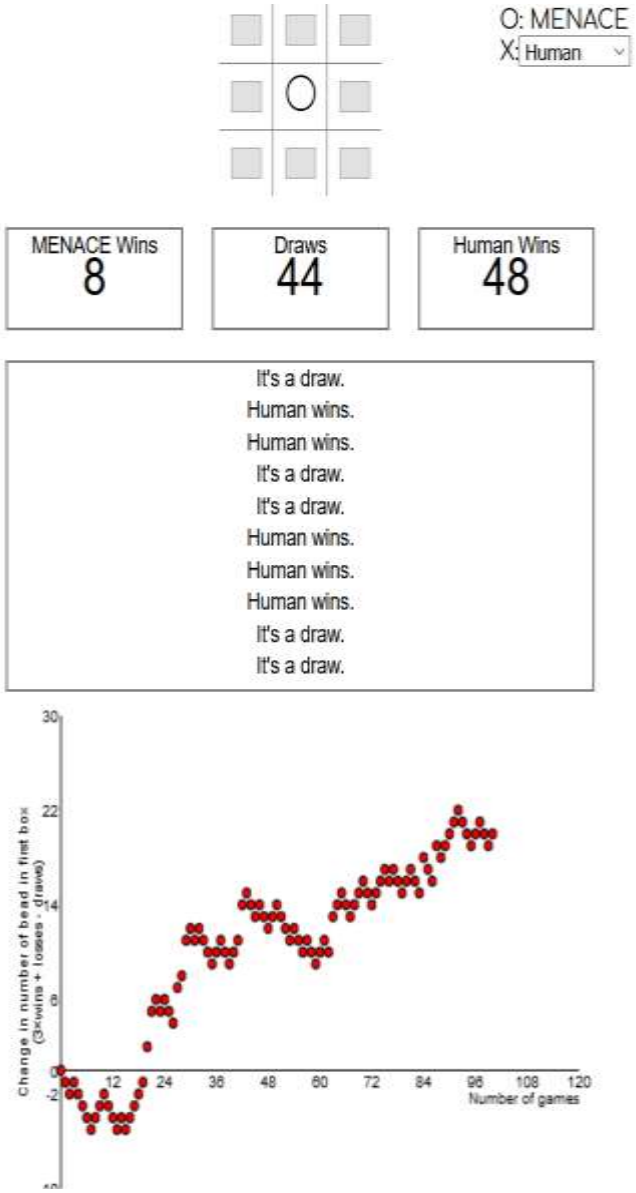


FIGURE 3:IMPROVEMENT IN MENACE PERFORMANCE SHOWN THROUGH THE GRAPH AFTER PLAYING 100 GAMES

After playing 100 games against MENACE, this is the final graph that I obtained.

This graph is similar to the original graph done by Donald Michie. You can see that even though the game starts on the negative Y-axis, as the game progresses it continues to stay on positive side of the Y-axis.

This indicates that MENACE's chance of losing is less compared to its chance of either winning or drawing as it plays lots of games.

1.2 STRUCTURE OF MENACE:

While explaining the game before we assumed that MENACE is made up of 304 matchboxes and proceeded with the game. Now let us try to understand why it has to be 304. If we go by common sense, the amount of match boxes to play Noughts and Crosses game would be very huge. And it would be difficult to construct it. So we use certain rules and assumptions, such that the structure of MENACE isn't too complicated.

For MENACE's first move, there is only one matchbox required since there is just an empty board pattern on it. For its second move, we need 72 matchboxes (i.e 9C1 * 8C1). Similarly for MENACE's third move, we need 756 matchboxes. For MENACE's fourth move, we need 1372 matchboxes. So to sum it up we will need 2201 matchboxes.

But if we observe carefully at the different possible layouts in the above 2201 matchboxes, we can see that some deigns are essentially same.

X	O	
	X	

	X	
X	O	

TABLE 3:BOARD CONFIGURATIONS DRAWN ON MATCHBOXES

The above two layouts are essentially same, second one is obtained by rotating the first one clockwise. So if the move chosen by MENACE is wrong in first layout, it should be able to rectify its mistake if it ever encounters the second layout in a game. So to help MENACE learn more effectively, it will be best for us to consider some changes in board to be same.

Donald Michie actually considered about eight different possibilities where we can find a similar layout. Like rotating the board clockwise by a quarter turn, by a half turn, by a three-fourth quarter turn, or the reflection of board on both vertical and horizontal axis, or the reflection against the two diagonal axes. It is okay for us to consider all the rotations and reflections of the board as same as one single layout, because all these positions essentially follow the same rules of a Noughts and Crosses game.

So considering the above mentioned possibilities, now the number of matchboxes reduces to 304. That is for MENACE's first move one match box, for

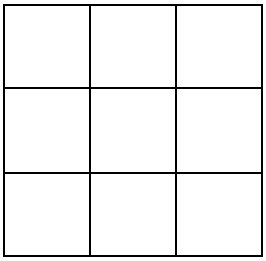
MENACE's second move 12 matchboxes, for MENACE's third move 108 matchboxes, and finally for MENACE's fourth move 183 matchboxes. So in total Donald Michie used 304 matchboxes.

Even though we made the structure a lot simpler by above assumptions, it would be difficult for the human to play the game now. Because now instead of directly spotting the layout, we have to search all the seven possible layouts (rotations and reflections) that are similar to the given layout. And we also have to relate the colour of bead to its correct position. Fortunately for us, this new burden of work is quite deterministic, and MENACE is also following strict rules with no possible space for ambiguity.

1.3 LEARNING ABILITY OF MENACE:

MENACE is learning what should be its action across the 304 different board configurations. MENACE will only get four moves to choose its action. And in its fifth move it can only make one action, which is left on the board. The possibility of MENACE making a winning move increases as the game progresses, due to the positive reinforcement (that is adding more beads which will result in winning). And similarly, the possibility of MENACE losing decreases due to negative reinforcement (that is removing the certain coloured bead which would result in losing).

Now to understand the performance of MENACE, we are in need of a parameter. As the number of the beads is the only variable thing in this game, we will consider that the number of beads of same colour in a single matchbox as a parameter. Let us take an example:



In this empty board, MENACE will make its first move. It will have 12 beads. That is four beads of three different colours. These three different colours represent, three different positions on board, centre of the board, middle of the edges or the corners. Since we consider all rotations and reflections on board as equal.

And if we consider MENACE's third move, each matchbox will have six beads. That is two beads of three different colours. These three different colours represent, again three possible positions on the board, one is a move in between the two X's, or in the middle of the left over two edges, or in the corners.

If we take the all kinds symmetric positions possible in the game with respect to these 304 matchboxes and differentiating the different moves from all the possible moves, we will find that there are total of 1720 beads. And 1087 parameters representing these beads.

To make learning easier for MENACE, Donald Michie made MENACE take the first move. Since we can observe that the first player in the tic-tac-toe game generally has an inherited bias to do better in game than the second player. MENACE usually plays uniformly randomly in all of its possible different moves against human. According to the research on tic-tac-toe games when two uniformly random players face each other, first player has a chance of 59% to win, and 13% chance to draw and 28% chance to lose. This clearly shows the bias towards the first player.

To make MENACE learn well we should carefully pick the moves against it, because how well the MENACE learns definitely depends on the player it is against. The perfect proof for this is the fact that Donald Michie saying MENACE doing better after only 220 games. Whereas other players required more than that 220 games for MENACE to play well. So we can assume that to make MENACE learn better Michie might have used a deliberately planned moves against it rather than some vaguely random moves. Michie has also said in his research paper that while making a computer simulation against MENACE, he found that randomly played moves resulted in a slower learning.

Even though MENACE can't do as well as human, we can try our best to make it play well by making it play against different kinds of opponents. The only bad thing about this process is that MENACE tries to tune itself to the opponent.

CHAPTER 2

2.1 MENACE AND MACHINE LEARNING:

We can observe that the way MENACE learns is entirely different from how a human learns. Because when a human first starts learning Noughts and Crosses game the first thing human makes sure is to complete three O's or three X's in a row. If we see MENACE learning, it never tries to complete the row.

	O	

		X
	O	

	O	X
	O	

	O	X
	O	X

TABLE 4:BOARD CONFIGURATIONS DRAWN ON MATCHBOXES

Now instead of completing the middle column this is what MENACE does in the computer simulation:

	O	X
O	O	X

		X
O	O	X
		X

TABLE 5:BOARD CONFIGURATIONS DRAWN ON MATCHBOXES

And thus resulting in MENACE losing against human.

Also while playing the game human learns that if he can't win, he can atleast block the machine from winning. He can do so by blocking three O's or three X's with his own move. But even then human has a possibility of losing if the

opponent has two continuous O's or X's that need to be blocked. He can only block one move and opponent wins in his next move.

These two above rules differs MENACE from human. The first rule is trying to get three continuous O's or three continuous X's. And the second rule is blocking the immediate win to opponent.

So instead of MENACE playing randomly, if we manage to incorporate these two rules, we can see a huge improvement in MENACE learning.

MENACE	PLAYER A	PLAYER B	PLAYER C
RANDOM	W:59%, D:13%,L:28%	W:0%,D:24%,L:76%	W:27%,D:19%,L:53%
RANDOM +TWO RULES	W:86%,D:10%,L:4%	W:0%,D:82%,L:18%	W:51%,D:37%,L:13%

TABLE 6:PERFORMANCE OF MENACE INCREASED AFTER ADDITION OF TWO NEW RULES

What is the difference between human and MENACE's ability? Why can't MENACE use the two rules that human can while playing the game?

It is because we as humans have some sense of geometrical positions on board, we know what three in a row or column or diagonal mean without even trying too hard. Whereas it is difficult for MENACE to get this point across. Because it can't differentiate different geometrical positions. Even if it did learn for one particular matchbox, it can't associate this with a matchbox with a slight change in configuration. It would be even harder for MENACE to understand vertical row, horizontal row and diagonal. Thankfully Michie helps MENACE understand geometrical configurations by making eight different possible rotations and reflections as one unique board position. But still at the end it is human (Michie) that is doing the thinking behind MENACE.

From the above paragraphs, we can see that MENACE learning is not similar to human learning. It is a different type of learning called Machine Learning. The term Machine learning isn't coined during the time MENACE was first created, it only came into light after MENACE's creation.

MENACE isn't aware about the fact that it is playing a game. It doesn't even know what the game is, it has its beads removed from matchbox by a human. And later based on positive or negative reinforcement the beads are either added back or removed. It is human (Michie), doing all the work. He is the one playing the tic-tac-toe game on the paper and he is the one deciding if the beads

should be added or removed. And he decides if the MENACE won or lost or draw the game. MENACE doesn't know that it is learning a game. It is simply following the instructions given to it by human. So MENACE is just a learning machine.

This is the case with Machine learning nowadays too. This fact could either be a plus point or negative point. Humans are the ones who makes the sketch of the real world problem and implement it on the machine. Humans give the input data to machine, make it respond to it by giving instructions to its learning system, and finally manage to give the desired output. Even though MENACE was found a long time ago, its basic structure still resonates with the modern days Machine learning.

We have said in the introduction part that the machine learning is divided into 3 different types, namely 1) supervised learning 2) unsupervised learning 3) reinforcement learning.

Which type do you think MENACE falls under?

The answer is Reinforcement learning. Because MENACE has received the feedback only after it has completed its given task. MENACE is one of the few earliest examples of reinforcement learning. It is the one that introduced the concept of reinforcement learning to people.

2.2 IMPLEMENTATION OF REINFORCEMENT LEARNING:

In this game MENACE makes three to four moves, and all this moves are done one after another in a row. How far should the feedback be traced back to? According to the original game done by Donald Michie, he considered to apply all kinds of reinforcements (positive or negative) to all of its moves. Though it is still uncertain how Michie reached this conclusion. Giving reinforcement at the final move is understandable, since it is where the game result is decided. But giving the same reinforcements to earlier moves is a little uncertain.

Let us first understand what reinforcement learning is.

REINFORCEMENT LEARNING: It is one of the methods implemented in Machine Learning. Reinforcement learning is trial and error type of learning. In other types of Machine learning, machine is usually presented with a large set of training data, and machine uses this training data to train itself. Whereas in reinforcement learning, machine isn't presented with any prior data. Reinforcement agent decides what action to take at one particular situation by trying to maximise the immediate reward as much as possible. That is machine

is learning here from its experience rather than the training data. Based on the output, reinforcement agent decides to either reward the machine or punish the machine.

This way when the process is repeated the probability of machine making a mistake reduces. This is how the training through reinforcement learning goes.

Let us see how we can compare reinforcement learning to MENACE's learning. Reinforcement learning is mainly used when there are finite number of states present. Here in MENACE, the number of uniquely different matchbox configurations (304) are used to represent states in reinforcement learning. All these 304 matchboxes represent the possible board layouts when it is MENACE's turn to play. For each of these 304 matchboxes (states), there are certain actions we need to take. These actions are represented by 9 different coloured beads, with each bead representing different possible move on the board. In reinforcement learning we have a policy which is the probability of action in its current state. Now applying this to MENACE, the policy would be total number of same colour beads in a matchbox to total number of beads in that same matchbox. And our goal is obviously to learn the policy which is best for MENACE at each state. And maximise it. It should also be noted that in MENACE it is difficult to return to back to the state once it has left, unlike in other games.

Even under reinforcement learning there are many methods you can use like Neural networks, Deep learning network, etc. But among these methods only one meets our requirement. That is Q-learning method. Neural network and deep learning network are not ideal for MENACE because they use certain function to represent policies of all states together. In MENACE we know the exact state we are present at each part of the game. We spread the feedback for winning, losing or drawing in the game equally among all of the MENACE moves. Though this is what Michie implemented it may not be ideal. Because a losing move at the final state need not necessarily imply that all the moves before it are useless. So to tackle with this problem, we use a new kind of learning method called as Q-LEARNING for MENACE.

In Q-learning, MENACE learns what the estimation of final reward is depending on its particular actions it has taken at each state. After learning about the approximate final reward, we then distribute the rewards among all moves.

Outlook for MENACE:

States: Board configuration on matchboxes, Actions: Number of beads of each different colour, Policy: Giving preference to the actions with most number of beads, Reward: Can be positive or negative based on the result of the game (win or lose or draw). And our goal would be to maximise the cumulative reward.

CHAPTER 3

3.1 RESULTS:

Let us try to implement reinforcement learning mathematically. How well we are able to optimise the mathematical equations decides how fast and efficient our program works.

Noughts and crosses game is obviously a sequential process. That is decisions at each state of the game are taken sequentially. Even though it is a sequential process it does reach an end with the MENACE either winning or losing or drawing in the game. Let us represent this outcome with a value.

Since MENACE learns through its experience rather than on training data. We try to give the actions which resulted in the final outcome some feedback (i.e. reinforcement). As we already know the reinforcement or feedback will be positive if MENACE has won or managed to draw the game.

What happens when we give some actions made by MENACE positive feedback? By giving positive feedback we are encouraging the MENACE to take the similar action the next time it encounters the same situation, since this action has given a positive result for MENACE. In the same way if some action has given a negative result, we give that action negative feedback so that it won't be encouraged in the coming future.

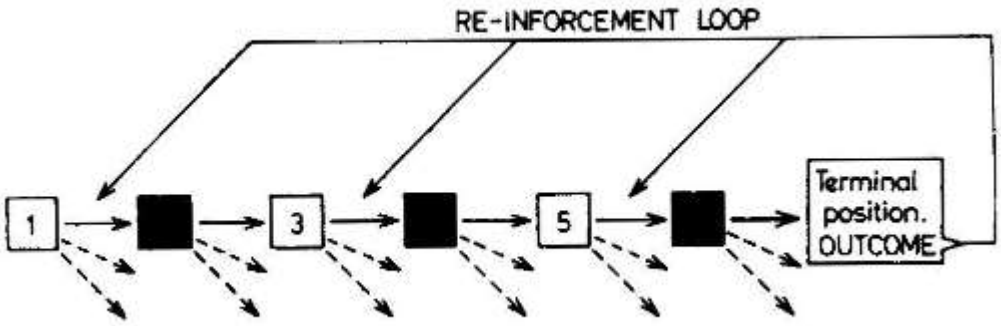


FIGURE 4:REINFORCEMENT LOOP FOR MENACE

This reinforcement loop represents how MENACE works. Since MENACE starts the game, the boxes 1,3,5 represents the state of MENACE and the arrows coming out of boxes represents the moves that MENACE can make. Based on the outcome we alter the previous moves. We add the feedback (positive or negative) back to the loop.

Reinforcement learning consists of a learning agent which decides what action we should take at any one particular state based on the policy. By choosing these actions, reinforcement learning agent makes sure to optimise the cumulative addition of all rewards at all possible states. That is agent doesn't only think about optimising its next reward, but it thinks about the action which will benefit on a whole.

Now implementing this to MENACE. For MENCAE 'i' is 304 i.e the number of match boxes. States Si represents various possible board configuration on matchboxes. Action is the number of beads of each of the nine different colours. The policy we decided to follow from the above paragraphs is to maximise this ratio (number of beads of a certain colour to the total number of beads present in the 'i'th matchbox).

MATHEMATICAL FORMULATION:

Let us now put all our explanations above in a mathematical formula:

States- Si belongs to S, 'i' belongs to (1,….,304), S is set of all possible board configurations. In other words, number of matchboxes

Actions-Ai belongs to A, 'i' belongs to (1,….,304), A is set of all possible actions possible at a given state. In other words, number of beads of different colours.

Policies-πi belongs to π, 'i' belongs to (1,….,304), π is mapped from set S to A. Policy is ratio of number of beads of a certain colour to the total number of beads present in the 'i'th matchbox.

Reward-function =R(Si,Ai), R is mapping SXA to real numbers. Reward is based on game won or lost.

Michie followed this particular reinforcements in each case. If MENACE wins add 3 more additional beads of same colour to the matchbox, if MENACE loses remove that bead of same colour from the matchbox, if MENACE draws the game return the bead of same colour to the matchbox.

And we will also consider a state transition function. Let us call it K(Si,Ai). K is mapped from SXA to S(since it is transition between states).

Now the reinforcement learning agent decides the Ai based on the πi. After the decided action by MENACE, reinforcement learning agent is put into a new state

based on K(Si,Ai). And agent also gets feedback from the reinforcement loop as shown in the Figure 4.

Now the reinforcement learning agent's goal is to maximise the cumulative reward as a whole, by choosing the right actions.

That is maximising $Ri = \sum_{j=i}^{\infty} \gamma^{j-i} Rj$, the cumulative reward. γ belongs to [0,1] and is called discount factor. Cumulative reward is addition of all rewards from the current state to final state of the process. Here final state is when MENACE won or lost.

While finding cumulative reward, we give weightage to all rewards based on the discount factor. Discount factor as the name suggests gives discount to the rewards which are far into the future.

Now since we have decided to use Q-learning, we need to understand what value-based learning mean, since Q-learning is part of it. How do we measure value of each state? In value-based method, value of each state is found by averaging the cumulative scores of that particular state. Hence value of each state is directly proportional to its rewards.

What we does in value based learning method is optimising value function with respect to the policies. Let V(S) be a value-function.

$V^{\pi}(Si) = \sum_{j=i}^{I} E_{\pi 0}[\gamma^{j-i}(R(Sj, Aj)|Si]$, I=304

E is expectation. Now we need to find a policy π which will help us maximise the expected cumulative rewards. This can be written in an equation as:

$V^{*}(S) = max_{\pi} E[\sum_{i=0}^{I}[\gamma^{i}(R(Si, Ai, Si+1)|\pi, S0 = S]]$, I=304

Now let's find optimal Q-function based on the $V^{*}(S)$.

Definition of optimal Q-function: we know that Q-function depends on both state and action.

Q(S,A)= R(S,A)+ $\gamma V^{*}(S')$, where S'=K(S,A) where K is state transition function.

$V^{*}(S) = max_{a} Q(S, A)$

Now combining the above two equations, we get :

Q(S,A)= R(S,A)+ $\gamma max_{a'} Q(S', A')$

That is Q-function of state S, depends on Reward for state S, for a given action A. And also on the discounted factor multiplied by maximum possible Q-function for the next state S' and a given action A'

Q-LEARNING METHOD

Now that we got to know what is Q-function. Let us try to understand what Q-learning is, and how we can use this in case of tic-tac-toe.

States/Actions	A1	A2	A3	A4	……
S1	Q(S1,A1)	Q(S1,A2)	Q(S1,A3)	Q(S1,A4)	……
S2	Q(S2,A1)	Q(S2,A2)	Q(S2,A3)	Q(S2,A4)	……
S3	Q(S3,A1)	Q(S3,A2)	Q(S3,A3)	Q(S3,A4)	……
S4	Q(S4,A1)	Q(S4,A2)	Q(S4,A3)	Q(S4,A4)	……
……	……	……	……	……	……

TABLE 7:Q-VALUES CORRESPONDING TO EACH STATE AND ACTION

If we look at the table for each state and action, we have corresponding Q-value. And this Q-value is important for us because it helps us decide if certain action at current state is good or bad.

We need to iteratively update the Q-values present in the above table. This is how Q-learning updates the Q-values:

Q_{i+1}(Si,Ai)=(1-α) Q_i(Si,Ai)+ α(Ri+ $\gamma max_{\mathrm{a}} Q_i(Si+1, A)$)

α is learning rate. α is a value between 0 and 1. If we look at the above equation carefully we can see that it is exponential moving average.

Since Noughts and Crosses game is quite a simple process, we need not go as far as to use moving average (this will be useful larger data problems to get a certain stability).

MENACE wins in the game. So assuming rewards for the game are +1 for winning, 0 for drawing and -1 for losing. We will now need to update the Q-values. We can take any random value for α, but let us take 0.9.

Assume all Q-values initially are zeroes. And now with the help of α and reward we will update the Q-values we have encountered during the course of the game.

O		

O		
	X	

O	O	
	X	

O	O	
X		
	X	

O	O	O
X		
	X	

TABLE 8: BOARD CONFIGURATIONS DRAWN ON MATCHBOXES

In the above game MENACE is the one that plays O, whereas X is played by human. Since MENACE won at the end of the game, it gets a reward of +1, so Q-value is updated to 0(initial value)+0.9(learning rate)*(+1). That is Q-value of third move of MENACE (also final move) is updated to be 0.9.

Now we can update Q-value of the previous step of the game by tracing our game back. The Q-value for the previous state would be 0(initial value)+0.9(learning rate)*0.9(maximum possible Q-value). Here we didn't multiply with the reward since this move is not the final move. And we have also considered discount factor to be 1. That is Q-value for the second move of MENACE is now updated to be 0.81.

Observe that we are updating the Q-values only at the positions on board where MENACE has made its move.

Now doing similarly for the first move made by MENACE, Q-value for it would be 0+0.9*0.81. That is 0.729.

Continue to repeat this process by playing as many games as possible. We can also play against different opponents.

To make MENACE best at noughts and crosses games, we actually should train it against lots of players whose strategies could be either random moves or moves that are trained from learning agents. And also we can vary the values of learning rate, discount factor and observe the efficiency of MENACE. By making MENACE play more than 500 games, it slowly starts to improve.

Finally by keeping table of performances of MENACE against all the variables mentioned above, and across thousands of games and tens of players, we can find one too many processes which improves MENACE efficiency.

3.2 CONCLUSION:

Donald Michie has introduced us to a new form of machine learning through noughts and crosses game. Though the learning method he has used for MENACE is quite efficient, it takes MENACE a long duration to improve in the game and it is also physically exhaustive to follow his method. Now with the improved technology and also with the new found knowledge of various types of machine learning, we are able to improve the MENACE ability. Machine learning has numerous applications in the real world. The applications of Machine Learning only seems to be increasing day-by- day. So it is fun to learn about the one of the basic starts to Machine learning through this project.

3.3 REFERENCES:

- http://www.mscroggs.co.uk/blog/19
- http://www.ccs.neu.edu/home/rplatt/cs7180_fall2018/slides/intro_rl.pdf
- https://people.csail.mit.edu/brooks/idocs/matchbox.pdf
- https://www.ke.tu-darmstadt.de/lehre/archiv/ws1011/ki/reinforcement-learning.pdf
- http://rodneybrooks.com/forai-machine-learning-explained/
- https://nestedsoftware.com/2019/07/25/tic-tac-toe-with-tabular-q-learning-1kdn.139811.html
- http://www.mathrec.org/old/2002jan/solutions.html