International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
***Article Received:** 26 August 2023 **Revised:** 20 October 2023 **Accepted:** 02 November 2023*

Understanding the Order of 500 and 1000 Rupees Notes Ban using Reinforcement Learning

Varsha D. Jadhav[1], Dhananjay R. Dolas[2], Gitanjali B. Yadav[3], Monali R. Borade[4], Ashwini R. Nawadkar[5]

[1]Artificial Intelligence and Data Science Department
Vishwakarma Institute of Information Technology
Pune, Maharashtra, India
e-mail: drvarshajadhav22@gmail.com

[2]Mechanical Engineering Deparment
Jawaharalal Nehru Engineering College, MGM University
Aurangabad, Maharashtra, India
e-mail: ***drdolasjnec@gmail.com*.**

[3]Artificial Intelligence and Data Science Department
Vishwakarma Institute of Information Technology
Pune, Maharashtra, India
e-mail: gitanjali.yadav@viit.ac.in

[4]Artificial Intelligence and Data Science Department
Vishwakarma Institute of Information Technology
Pune, Maharashtra, India
e-mail: monali.borade@viit.ac.in

[5]Artificial Intelligence and Data Science Department
Vishwakarma Institute of Information Technology
Pune, Maharashtra, India
e-mail: ashwini.nawadkar@viit.ac.in

Abstract— In the field of machine learning called reinforcement learning, complicated sequential decision-making problems have been addressed. The issue that arises when an agent learns behavior by trial-and-error runs to determine the ideal policy, or the sequence of behaviors so that rewards are maximized,is known as reinforcement learning. Because many reinforcement learning methods use dynamic programming approaches, the environment is characterized as a Markov Decision Process (MDP). The research presents reinforcement learning using Bigram, trigram, and 4-gram models for tweets collected for "500 and 1000 notes banned." A multistage graph problem is used to draw the graph and the Bayes method is used to compute the probabilities. For the given word sequence, it determines the shortest route between source and destination. After that, the path is defined by the agent's randomly selected states and actions, which are subsequently followed to receive rewards. Epsilon greedy selection mode randomly chooses an action to explore the environment.

Keywords-Reinforcement Learning, Markov Decision process, agent, environment, epsilon greedy.

I. INTRODUCTION

Reinforcement learning refers to the problem of an agent attempting to choose the best course of action through trial-and-error connections with a changing environment. All reinforcement learning strategies have the same feedback limitation on the agent: a reward signal showing how well the agent is behaving. In contrast to supervised machine learning techniques, there is no direction on how to improve its behavior. The goal and challenge of reinforcement learning [1] is to improve an agent's behavior while using only this limited sort of input. A decision-making agent, the learner in reinforcement learning makes actions in the environment and is rewarded for doing so in an effort to solve a problem. The best policy, defined as a set of actions that maximizes overall reward, should be identified through a series of trial-and-error runs. A decision-maker known as an agent is positioned in an environment. The environment is always in a specific condition, one of a range of potential states. The decision-maker conjures up a conceivable set of activities. A series of activities are required to complete the task, and feedback in the form of a reward is received. The reward describes the problematic and is required if we want a learning agent to determine the best sequence of actions to take to resolve it, where "best" is defined as the set of steps that will result in the highest potential rise in rewards over time [2]. The agent's interaction with the environment is depicted in Fig. 1. The agent acts in any environment condition to bring about a change and produce a reward.

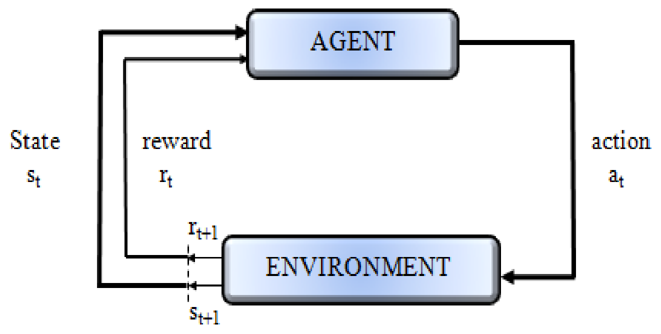

Figure. 1The agent interacts with the surrounding environment.

In a reinforcement learning problem, the intermediate states and actions are taught to generate an internal value based on how well they will help the learner achieve their objectives and receive the desired rewards. Once the agent has mastered such an internal reward mechanism, he or she need only do the appropriate local action to maximize it. A series of actions that were employed earlier in the Markov model are needed to complete the task. The order of events is displayed using a multistage graph problem. From the begin node to the target node, the graph is divided into stages. The Markov model varies in that the action sequence is generated by an external process. However, in the case of reinforcement learning, the agent creates the action sequence. When an agent does not know its whole state but must deduce it with some doubt from observations, a partly observable Markov decision process may be utilized in reinforcement learning. The problem requires to find the shortest transformation sequences from the start word to the end word. The shortest path can be found out using bi-gram, which is one word followed by another word. Also, in tri tri-gram, one word followed by two words, and in 4-gram, one word followed by three words. N-gram can also imply. Breadth fist search (BFS) is already existing method. The research work tries to solve the problem using reinforcement learning which is artificial intelligence technique that can help to improve vocabulary as well as spellings and phonic skills.

II. LITERATURE SURVEY

Introduction to reinforcement learning by Amman et al. [3] illustrates the logic behind reinforcement learning. The details for resolving reinforcement problems are also described. They talk about the potential and difficulties of reinforcement learning. From a computer science perspective, Leslie Pack, et al. [4] review the origins of reinforcement learning as well as some recent research. A comprehensive survey of current developments in reinforcement learning were made by Abhijeet Gosavi [5]. The survey provides more than 100 citations from the body of knowledge to spark fresh research ideas. From a machine learning standpoint, Samiksha Mahajan [6] presents an introduction of reinforcement learning and examines the fundamental concepts and methods that are applied to solve reinforcement learning issues. A hybrid approach was given by Pieter Abbel et al. [7] and just needs a rough model and a few real-world trials. The algorithm can achieve close to ideal performance in real systems, according to empirical studies, when given a basic model and a few real-world trials. By assuming an agent acts on the input and learns value functions, Macro, et al. [8] present a new framework for using reinforcement algorithms to solve classification tasks. It also explains how classification Markov decision processes can be used to model classification problems and presents the Max-Min ACLA algorithm, an expansion of the innovative reinforcement learning algorithm.

III. MARKOV DECISION PROCESS

The mathematical outline used to define a explanation to a reinforcement learning problem is known as the Markov Decision Process [9]. This might be done as follows:

- A group of states, S
- A group of actions, A
- Reward function, R
- Policy, π
- Value, V

To move from the initial state to the end state (S), an action (A) is taken. Rewards (R) are received in exchange for every activity taken. These behaviors produce either favorable or unfavorable results. Policy is defined by the acts taken, while value is defined by the incentives received in return. By selecting the ideal course of action for each potential value of S over the course of time t, the objective at hand is to maximize the rewards.

$$E(r_t \mid \pi, s_t) \tag{1}$$

IV. REINFORCEMENT LEARNING COMPONENTS

The agent is the name given to the learning decision-maker. The environment rewards the agent when they do a certain action. As t = 0, 1, 2..., time is discrete, and $s_t \in S$ signifies the agent's state at time t, where S is the set of all possible states. (s_t) signifies the set of feasible actions in state s_t, and a_t symbolizes the action the agent takes at time t. When the agent in state s_t performs the action a_t, the timer starts, the agent receives reward $r_{i+1} \in R$, and then the agent advances to state s_{t+1}. The Markov Decision Process (MDP) is used to model the problem. The reward and the next state are drawn from their corresponding probability distributions $(r_{t+1}|s_t, a_t)$ and P $(s_{t+1}|s_t, a_t)$. The Markov system is used here, and the state and reward in the resulting time step are determined only by the present state and action [10].

Depending on the application, a specific state may be set as the original state, and some requests additionally have a goal state where the search is complete. All actions in this goal state have a chance of one and receive no reward. The succession of acts from the beginning to the target condition is referred to as an episode or trial.

The policy, Π, describes the agent's performance and is a mapping from the states of the environment to actions: Π: s → λ. The policy describes the action to be taken in any state s_t : a_t= Π (s_t). The value of a policy

Π, $V^{\Pi}(s_t)$, is the predictable increasing reward that will be acknowledged while the agent follows the policy, opening from state s_t.

The agent in the episodic or finite-horizon model seeks to maximize the predictable reward for the next T steps:

$$V^{\Pi}(s_t) = E[r_{t+1} + r_{t+2} + ... + r_{t+T}] = E\left[\sum_{i=1}^{T} r_{t+i}\right] \quad (2)$$

There is no predetermined end to the episode, but some duties are still ongoing. There is no sequence limit in the infinite-horizon concept, but future benefits are discounted:

$$V^{\Pi}(s_t) = E\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ...\right] = \left[\sum_{i=1}^{\infty} \gamma^{i-1} r_{t+i}\right] \quad (3)$$

where $0 \le \gamma < 1$ is the discount rate to keep the return finite. If γ =0, then only the fast reward counts, As γ approaches 1, rewards further in the future count more, and we say that the agent becomes more far-seeing. γ is less than 1 because the series of steps required to perform the assignment typically have a time constraint. Given the uncertainty surrounding its survival, rewards are recommended as soon as possible. For each policy Π, there is a $V^{\Pi}(s_t)$, and we want to find the optimal policy Π^* such that

$$V^*(s_t) = \max_{\Pi} V^{\Pi}(s_t), \forall s_t \quad (4)$$

In some applications, for as an alternative of working with the values of states, V(s_t), the values of state-action pairs, Q (s_t, a_t),V(s_t) denotes how good it is for the agent to be in state s_t , whereas Q(s_t,a_t) denotes how god it is to perform action a_t when in state s_t and then conforming the best policy afterward. The value of a state is equal to the value of the best likely action:

$$V^* = \max_{a_t} Q(s_t, a_t)$$
$$= \max_{a_t} E\left[\sum_{i=1}^{\infty} \gamma^{i-1} r_{t+i}\right]$$
$$= \max_{a_t} E\left[r_{t+1} + \gamma \sum_{i=1}^{\infty} \gamma^{i-1} r_{t+i+1}\right]$$
$$= \max_{a_t} E\left[r_{t+1} + \gamma V^*(s_{t\_1})\right]$$

$$V^*(s_t) = \max_{a_t}\left(E[r_{t+1}] + \gamma \sum_{s_{t-1}} P(s_{t+1} \mid s_t, a_t) V^*(s_{t+1})\right) \quad (5)$$

Each possible next state s_{t+1} is reached with the probability P(s_{t+1}|s_t, a_t), and continuing from there using the optimal policy, the expected cumulative reward is $V^*(s_{t+1})$. All possible next states are summed over, and discounted because it is one time step later. Adding immediate expected reward, the total expected cumulative reward for action a_t is obtained. Then choose best of possible actions. Equation (5) is known as Bellman's equation. Similarly,

$$Q^*(s_t, a_t) = E[r_{t+1}] + \gamma \sum_{s_{t+1}} P(s_{t+1} \mid s_t, a_t) \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1})$$

(6)

Once $Q^*(s_t, a_t)$ values are obtained, then the policy π is taken as a_t^* ,which has the highest value among all $Q^*(s_t, a_t)$:

$\pi^*(s_t)$: choose a_t^* where

$$Q^*(s_t, a_t^*) = \max_{a_t} Q^*(s_t, a_t) \quad (7)$$

This means that if $Q^*(s_t, a_t)$ obtained, then by using epsilon greedy search at each local step, the optimal sequence of steps that maximizes the cumulative reward, are obtained.

V. DEEP Q-LEARNING

A model-free reinforcement learning method is called deep Q-learning. A policy-based learning algorithm called Q-learning uses a neural network as the function approximator. It can be used to determine the best Markov Decision Process (MDP) action-selection policy. It functions by teaching the user an action-value function, a state-action pair function that predicts how effective a specific action will be in a specific condition. In order to precisely identify an activity that will offer the best total reward after being in that particular condition, the value-function is estimated [11]. A policy is a guideline the agent adheres to while deciding which actions to do in light of the condition it is in. After learning such a value-action function, the ideal policy can be produced by simply selecting the action with the highest value at each level. One of the benefits of Q-learning is that it may compare the expected utility of various actions without the need for an environment model. Furthermore, Q-learning can handle problems including stochastic transitions and rewards without the need for adjustments. It has been demonstrated that Q-learning eventually discovers an optimal policy for any finite MDP, which means that the expected value of the total reward return across all subsequent steps, beginning with the present state, is as high as it can be.

A. Algorithm

A collection of actions *A* per state *S*, and an *agent* make up the model. The agent can do the transition from one state to the next by performing an action, or A. An agent receives a reward in the form of a score after performing an action in a particular condition. The agent seeks to maximize its overall return. It achieves this by discovering the best course of action for every state. The course of action with the biggest potential long-term return is the one that is best for each state. The expected values of the prizes for all upcoming steps starting from the current state are added up to form this award. The expected value is the long-term median of all iterations of the experiment [12].

The procedural form of algorithm is:

```
Step1:  Initialize Q(s,a) arbitrarily where s ε S and a ε A
Step 2: Repeat (for each episode):
Step 3: Initialize S
Step 4:   Repeat (for each step of episode):
Step 5:  Choose a from S using policy derived from Q
    // Using epsilon –greedy
Step 6:   Take action a, observe r, s'
Step 7:   $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{\alpha} ,Q(s',a') - Q(s,a)]$
Step 8:   $S \leftarrow s'$;
Step 9:   Until s is terminal
```

The Q-value table is initialized (s, a) in the algorithm above. S is the present state that is seen. Based on the action-selection policy (epsilon-greedy), an action is selected for the state. Following the action, rewards are received, and the new state s' is obtained. The supreme reward that can be given for the following state is updated together with the Q-value for the state using reward. The algorithm's formula and parameter descriptions are followed when updating. Up until the terminal state is reached, the process is repeated after the new state has been set.

The parameters used in the Q-value update procedure are:

a) α – the learning rate, set between 0 and 1. If it is set to 0, no learning occurs because the Q-values are never updated. A high score, like 0.9, indicates that learning can happen quickly.
b) γ - discount factor, also set between 0 and 1. This illustrates the idea that benefits in the future are worth less than rewards today. For the process to converge mathematically, the discount factor must be set lower than 0.
c) $\max_{\alpha}$ - the maximum reward that is attainable in the state following the current one i.e. the reward for taking the optimal action thereafter.

The flowchart for Q-learning is as shown in fig.2.

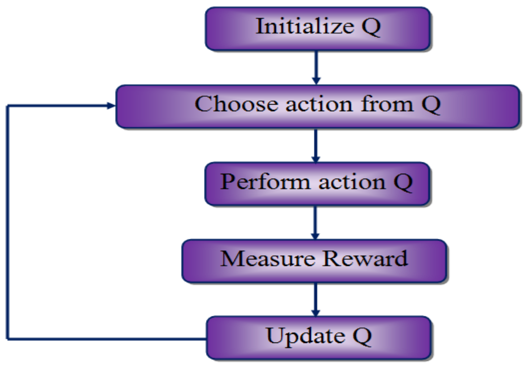

Figure 2. Q-Learning

VI. DATA RETRIVAL

The information was obtained using the Twitter API. The Prime Minister enacted a ban on 500 and 1000-dollar bills on November 8th, 2016 in the evening. From November 9 until November 14, 2016, tweets were gathered. “Banned 500 and 1000 notes" is the keyword. 42,000 tweets in all were gathered over the course of six days. Only three tweets, out of 42000 total, are thought to demonstrate the bi-gram, tri-gram, and 4-gram models.

1. Rs 500 and Rs 1000 notes
2. 500 and 1000 Rs notes are banned.
3. Rs 500 and 1000 notes banned in India.

After pre-processing the tweets will be

1. rs and rs notes
2. and rs notes are banned
3. rs and notes banned in india

Using the Bayes approach, which determines probabilities and plots a graph from source to destination, bi-gram, tri-gram, and four-gram models were created. A state is represented by each circle. The likelihood of changing states is represented by each edge. The transition is impossible if this value is 0, certain if it is 1, and may be somewhere in between if it is 0. It is a potential word order in language models if we trace with our finger every route from each state to each accepting state.

VII. REINFORCEMENT LEARNING MODELS

The reinforcement learning models are build for bi-gram, trigram and 4-gram models.

A. *Reinforcement learning for bi-gram Model*

The probability of the start word and one word followed by another word are computed for the purpose of generating bi-gram graphs. Naive Bayes method is used to calculate the probability. First the probability of start words is calculated. The start words in the above three tweets is *rs, and*. The probability of the start words is computed by dividing the total number of that word occurring as first word divided by the total occurrences of the same word. For example, *rs* is start word 2 times, in tweet 1 and tweet 3. The total number of times *rs* occurring is 4. So, probability of *rs* as start word is P(rs|<s>) =

2/4=0.5. Similarly, probability of *and* as first word is P(and|<s>) = 1/3 =0.33
When the preprocessed tweets are observed, it is found that in first tweet *rs* is followed by *and* and in the second tweet *and* is followed by *rs* which means that it is looping. Hence the bigram sequence considered and its respective calculated probability is as follows:

a) rs followed by notes. P(notes|rs) = 2/4 =0.5
b) notes followed by are. P(are|notes) = 1/3 =0.33
c) and followed by notes. P(notes|and) = 1/3 =0.33
d) notes followed by banned. P(banned|notes) = 1/3 =0.33
e) banned followed by india. P(india|banned) = 1/3 =0.33

A graph is drawn using the probabilities and then the multistage graph problem is used to compute the shortest path from the start word to the end word. The number inside the node indicates the node number and the graph is divided into stages from V1 to V5. There are seven nodes and the graph is having five stages. The cost from the start node to the end node is calculated using a forward method using the equation below.

$$cost(i,j) = \{C(j,l) + cost(i+1,l)\} \qquad (8)$$

The minimum cost path is traced out from node 1 to 3, to 4, to 6, to 7. The sequence of words traced out is *and notes banned india* which clearly indicates that notes were banned in india. The cost computed is 1.32 and the path is shown in red edges in figure 3.

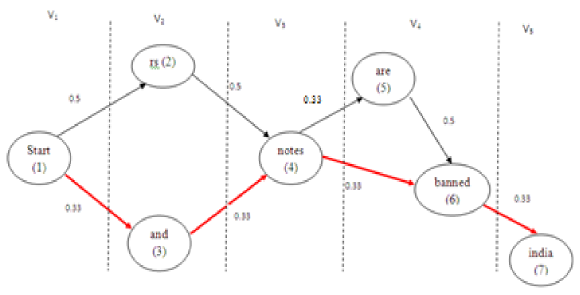

Figure 3 Minimum Cost Path for bigram graph.

A reward is encountered as it transits from one state to the next and one of the possible actions—move up, down, left, or right—is selected at each stage.

The traveling salesperson dilemma is depicted on the graph. The goal is to go from state 1 to state 7 for the least amount of money possible. The distance traveled at each edge between two nodes is indicated by a number. Cost is some money earned along the road. Value is outlined as the overall cumulative reward following the completion of the policy.
The set of states are the nodes 1,2,3,4,5,6,7

- The action to taken is to go from one node to another.
- The reward function is the value represented by edge which is the cost.
- The policy is the way to complete the task.

Now suppose the start state is state (1), the only visible path is the next destination and anything beyond that is not known at this stage. Greedy approach can be followed and next possible step is taken, which is going from state (1) to state (3) from a subset of {1→(2,3)}. Now state 3 is reached and wants to go to state 7, so 3→ 4 is chosen. Now state (4) is reached and want to go to state 7, so {4→(5,6)} can be chosen. 4→ 6 have lowest cost and hence the path is taken. Now state 6 is reached and wants to go to the destination node (7), so 6→7 is reached. So, the policy was to take {1→ (3, 4, 6, 7)} and the value is 1.32. This is known as epsilon greedy, which is a greedy method to problem resolving. Now if again state (1) to state (7) is to be reached, then the same policy is followed.

B. *Reinforcement Learning for Tri-gram Model*

For building the tri-gram model, the probabilities of start word is calculated and by moving from one word followed by two words. Initially the probability of start word is calculated which is P(rs|<s>) = 2/4 = 0.5 and P(and|<s>) = 1/3 = 0.33.

Now the probability of one word followed by two words is calculated as follows:

a) rs followed by and,rs. P(and,rs|rs) = 2/4 =0.5
b) and followed by rs,notes. P(rs,notes) = 2/4 =0.5
c) rs followed by notes, are. P(and, rs|rs) =2/4 =0.5
d) rs followed by notes are. P(notes, are|rs) = 2/4 = 0.5
e) notes followed by are, banned. P(are, banned|notes) = 1/3 = 0.33
f) are followed by banned, india. P(banned, india|are) =1/3 = 0.33

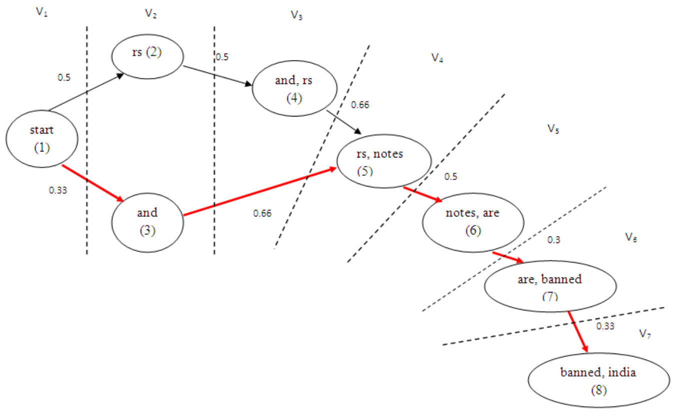

Figure.4 Minimum cost path for Tri-gram graph

C. *Reinforcement Learning for 4-gram Model*

For building the 4-gram model, the probabilities of start word is calculated, and moving from one word followed by three words. Probability of start word *rs* P(rs|<s>) = 2/4 = 0.5, P(and|<s>) = 1/3 = 0.33. The probability of one word followed by three words is as follows:

a) rs followed by and,rs,notes. P(and,rs,notes|rs) =2/4 = 0.5
b) and followed by rs,notes,are. P(rs,notes,are|and) = 2/3 = 0.67

c) rs followed by notes,are,banned). P(notes,are,banned|rs)=1/4 =0.25

d) notes followed by are,banned,india. P(are,banned,india|notes) = 1/3 =0.33

Applying multistage graph problem, we have stages V_1 through $V_{6.}$ The minimum cost path is 1.58

The epsilon greedy approach is followed and the policy is to take from {1, →(3, 5, 6, and 7)} and the value is 1.58.

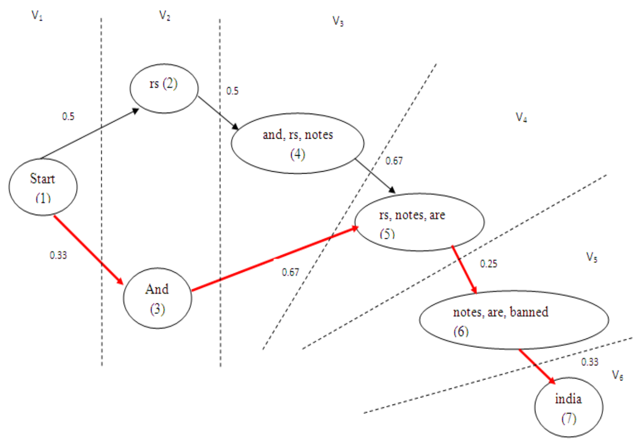

Figure 4. 4-gram graph

VIII. RESULTS AND DISCUSSION

A reinforcement learning agent learns new skills by trying with different state, action, and reward sequences. The option 'data' must be a data frame object, with each row indicating a transition tuple (s, a, r, s_new). The column names for each of the individual tuple members must also be included in 'data'.

A state-action table and an optimal policy that specify the appropriate course of action in each state are the outcomes of the learning process. One can quickly validate the agent's performance by specifying an environment function to model the dynamics of the environment. Apply the learnt policy to recently created samples to achieve this. The epsilon-greedy technique is employed for this. This method involves the agent choosing an action with probability at random while it explores the environment. As an alternative, the agent makes the best decision with probability 1 - by taking advantage of its existing information.

Table 1 displays the example Experience and state-action for bi-gram, tri-gram, and 4-gram using epsilon greedy action selection which is reinforcement learning method.

TABLE I. SAMPLE EXPERIENCE FOR BI-GRAM,TRI-GRAM,4-GRAMUSING EPSILON GREEDY METHOD

State	**Action**			**Reward**			**Next State**		
	Bi-gram	***Tri-gram***	***4-gram***	***Bi-gram***	***Tri-gram***	***4-gram***	***Bi-gram***	***Tri-gram***	***4-gram***
S1	Down	Down	Down	-1	-1	-1	S3	S3	S3
S2	Right	Left	Right	-1	-1	-1	S2	S2	S2
S3	Right	Right	Right	-1	-1	-1	S4	S5	S5
S4	Up	Left	Left	-1	-1	-1	S6	S4	S4
S5	Right	Down	Down	-1	-1	-1	S6	S6	S6
S6	Up	Right	Right	10	-1	10	S7	S7	S7
S7	down	up	down	-1	10	-1	S7	S7	S7

Table 1 displays the agent's randomly selected action, which is either up, down, left, or right. Additionally, it displays the next state obtained and the reward received for bigram, trigram, and four-gram models. The target is to reach the end word. The reward -1 shows that it did not reach the target, whereas reward 10 indicates the targeted end word. The agent makes best decision based on its existing information. In bi-gram the start state is S1and upon down action it attends S3 state. Now from S3 it goes to S4, from S4 to S6, S6, to S7 where it wins a reward of 10 and concludes that it is the target end word. In the case of tri-gram and 4-gram, the start state os S1 and the next state is S3, from S3 it goes to S5, From S5 it goes to S6, and from S6 to S7 where it received a reward of 10 and wins the target end word. In case of tri-gram, during S6 to S7 transition the reward is -1 for Right action, which shows that it has not reached the end word. But upon up action it reaches the S7 state which is the target.

IX. CONCLUSIONS

The total number of states and actions is specified. Move left, right, up, and down are the actions. The agent uses the epsilon-greedy approach to select one of the four possible actions at random for each state. The agent receives compensation. It leads to a reward of -1 when it changes states. The agent receives a reward of 10 if it leads to the goal position. The complexity of BFS is O(N*L) time, where N is the number of words and L is the average word length. In the case of the epsilon greedy technique, the model generates a random number at each step. If the number is less than epsilon in that step, the model does a random action; if it is greater than epsilon, the model takes an action based on what has been learned. BFS and epsilon greedy approaches cannot be compared, although it is simpler than BFS because a reward is introduced, which aids in decision making based on available information. Although the starting point is not stated, the agent's journey begins in line with the set actions. A manual intervention is required. Improved artificial intelligence algorithms can be added to reduce or eliminate manual intervention.

ACKNOWLEDGMENT

The authors are very much thankful to the management of Vishwakarma Institute of Information Technology, Pune for providing with resources for the experimentation. Also, Thankful the management of MGM University, Aurangabad for their valuable guidance.

REFERENCES

[1] Kaelbling, L. P., Littman, M. L., Moore, A. W., 1996. Reinforcement learning: A survey. Journal of artificial intelligence research 4, 237–285. https://doi.org/10.1613/jair.301

[2] Sutton, R. S., Barto, A. G., 1998. Introduction to reinforcement learning. Vol. 135. MIT press Cambridgearni. https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf

[3] Ahmad Hammoudeh. A Concise Introduction to Reinforcement Learning,2018,Researchgate.net http://dx.doi.org/10.13140/RG.2.2.31027.53285

[4] Leslie Park Kaelbling, Micheal L. Littman, Andrew W. Moore, Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research 4, pp237-285. (c) 1996AI Access Foundation and Morgan Kaufmann Publishers. https://jair.org/index.php/jair/article/view/10166/24110

[5] Abhijit Gosavi. Reinforcement Learning: A Tutorial Survey and Recent Advances. INFORMS Journal of Computing. http://dx.doi.org/10.1287/ijoc.1080.0305

[6] Samiksha Mahajan.: Reinforcement Learning: A review from Machine Learning Perspective. International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4,Issue 8,August 2018.ISSN:2277 https://www.semanticscholar.org/paper/Reinforcement-Learning%3A-A-Review-from-a-Machine-Mahajan/c8716f2bea0a3b8e8b62ffe06908c96c861cbed7

[7] Pieter Abbeel, Morgan Quigley, Andrew Y.Hg. Using Inaccurate Models in Reinforcement Learning. Proceedings of the 23rd International Conference on Machine Learning, Pittsburg,2006, copyright 2006 by the authors. http://dx.doi.org/10.1145/1143844.1143845

[8] Marco A. Wiering, Hado van Hasselt, Auke-DirkPietersma and Lambert Schomaker. Reinforcement Learning algorithms for Solving Classification Problems.2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning,11-15 April 2011. https://doi.org/10.1109/ADPRL.2011.5967372

[9] Ethem Alpaydm: Introduction to Machine Learning, 2nd edition. The MIT Press Cambridge, Massachusetts London, England (2010). https://kkpatel7.files.wordpress.com/2015/04/alppaydin_machinelearning_2010.pdf

[10] Seyed Sajad Mousavi, Michael Schukat, Enda Howley. Deep Reinforcement Learning: An overview. Proceedings of SAI Intelligent Systems Conference,23 June 2018. http://dx.doi.org/10.1007/978-3-319-56991-8_32

[11] Hado van Hasselt, ArthurGuez and David Silver.Deep Reinforcement Learning with Double Q-Learning. Copyright © 2016, Association for the Advancement of Artificial Intelligence. https://doi.org/10.1609/aaai.v30i1.10295

[12] Tom M. Mitchell. Machine Learning. McGraw-Hill Science/Engineering/Math (March 1,1997) , ISBN:0070428077, pp. 367-388