International Journal of Scientific Research in Science, Engineering and Technology
Print ISSN: 2395-1990 | Online ISSN : 2394-4099 (www.ijsrset.com)
doi : https://doi.org/10.32628/IJSRSET

Design and Training of AI Agent using Deep Q -Learning and Carla

Bhavanisankari. S[1], Vimal William[2], Sachin. A[2]
[1]Associate Professor Department of Electronics and Commuication Engineering, Jerusalem College of Engineering, Anna Univesrity Chennai, India
[2]UG Scholar Department of Electronics and Communication Engineering, Jerusalem College of Engineering, Anna University Chennai, India

Article Info

Publication Issue :
Volume 10, Issue 1
January-February-2023
Page Number : 74-81

Article History
Accepted : 05 Jan 2023
Published: 19 Jan 2023

ABSTRACT
The construction of artificial intelligence agents for controlling the vehicles autonomously is computationally expensive and requires more hardware resources in training the agent. This research focuses on constructing a low-power agent which can be deployable in space-constrained devices. Carla is an open-source environment, that gives a simulation of real-world experiences especially in training the artificial agent with stochastic elements which is similar to the real human world. As the real-world training of the agent can be costly and requires more resources, simulation helps in creating an environment with fewer resources and results in better accuracy in the real-world deployment of the model. Deep Q – Learning is a form of reinforcement learning algorithm, that is used as the base for creating the agent. The DQL is an advanced version of Q – Learning which helps in handling the memory in training and testing the agent, as the Q – Learning requires a lot of memory.
Keywords – Artificial Intelligence, Deep Q – Learning, Carla, Computer Vision, Neural Network

I. INTRODUCTION

The development of an effective artificial intelligent agent that is capable of operating cars autonomously places a strong emphasis on elements like the environment in which the agent is educated and the underlying learning methods. Oscar [1] demonstrated a method to train the artificial intelligence agent using Carla. The method focused on Deep Learning (DL)-based algorithms are used in an autonomous vehicle's control layer. To compare outcomes between them, Deep Reinforcement Learning (DRL) algorithms such Deep Q- Network (DQN) and Deep Deterministic Policy Gradient (DDPG) are specifically implemented. The purpose of this work is to develop a trained model using a DRL algorithm that is capable of instructing the vehicle to navigate correctly and efficiently along a predetermined itinerary. Additionally, for each algorithm, a number of agents are suggested as a solution; each agent employs a different set of data sources to execute the vehicle control orders. An open-source simulator like CARLA is used for this, giving the system the option to run a variety of experiments risk-free in a highly realistic urban simulation environment

something that would be impossible in the actual world. The results demonstrate that both DQN and DDPG succeed in achieving the objective, although DDPG performs better. Syavasya [2] explained about the primary issue facing the machine learning mechanism at the core of the speed control mechanism is the main thrust area related to autonomous cars. The efficient algorithm to address the problems posed by autonomous vehicle driving and decision-making in complicated contexts is reinforcement learning (RL). Because it lowers risk and conserves resources, a simulative environment is advantageous for the training and validation of an RL algorithm. A unique hybrid algorithm made up of the Deep Reinforcement Learning (DRL)-stochastic algorithm-Deep Deterministic Policy Gradient (DDPG)-SHAPley Additive exPlanations (SHAP) is introduced in this study effort. The introduction of an RL environment for optimizing longitudinal control is the main goal of this study work. Wang
[3] purpose a deep reinforcement learning (DRL) based left- turn decision-making framework at unsignalized intersection for autonomous vehicles. The objective of the studied automated vehicle is to make an efficient and safe left-turn maneuver at a four-way unsignalized intersection. The exploited DRL methods include deep Q-learning (DQL) and double DQL. Simulation results indicate that the presented decision-making strategy could efficaciously reduce the collision rate and improve transport efficiency. This work also reveals that the constructed left-turn control structure has a great potential to be applied in real-time.

The above cited papers explain the major successive mythology in the area of developing an artificial intelligent agent using deep Q – learning as the base algorithm which combines with the deep learning to learn about the environment based on the earned rewards and punishments by the algorithm for each and every action. Further, the weights of the network are collected for the real-world deployment.

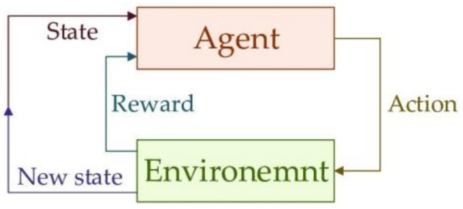
II. RELATED WORK

The design and train of the agent undergoes various stages and each stage developed with the technology that gives the model to understand the environment and acts accordingly. This section gives the theoretical concepts and mathematical support behind it.

A. Reinforcement Learning

The goal of reinforcement learning is to optimize a numerical reward signal by learning what to do and how to link situations to actions. The student is not given instructions on what to do; rather, they are left to experiment to determine which activities will result in the greatest rewards. The most intriguing and difficult situations are those in which choices can influence not only the immediate reward but also the next circumstance and, consequently, all subsequent benefits. Trial-and-error searching and delayed rewards are the two most crucial traits that set reinforcement learning apart from other types of learning. Matthew [4] mentioned in his paper about reinforcement learning that deep reinforcement learning (RL) techniques have propelled tremendous advancements in artificial intelligence, outperforming humans in games like no-limit poker, Go, and Atari. Cognitive scientists that are interested in comprehending human learning have been interested in this development. While the first wave of deep RL techniques did raise this issue, recent AI research has produced strategies that allow deep RL systems to learn faster and effectively. Meta-learning and episodic memory are two strategies that stand out as particularly intriguing and promising. Deep RL techniques that use episodic memory and meta-learning have immediate and intriguing implications for psychology and neuroscience in addition to their relevance as AI tools. These strategies bring to light a nuanced but crucially significant insight: the relationship between quick and slow kinds of learning is fundamental.

The type of learning that is examined in the majority of recent studies in the field of machine learning, supervised learning, is distinct from reinforcement learning. A knowledgeable external supervisor provides a training set of labelled examples for supervised learning. Each example includes a description of a circumstance as well as a label— the specification of the correct action the system should do in response to that situation—which is frequently to define a category to which the situation belongs.



The many stages of Reinforcement learning (RL) in the context of value-based learning are shown in Fig. 1. based onwhich the agent moves from one state to another, and in which a value is utilized to provide rewards and penalties.

B. Deep Learning

The concept of learning from examples is the foundation of the more comprehensive subject of artificial intelligence known as machine learning, which includes deep learning. In machine learning, we offer a computer a model to evaluate instances with and a limited set of instructions to adjust the model when it makes a mistake rather than giving it a lengthy list of rules to solve the problem. We anticipate that a suitable model would be able to handle the issue very precisely over time.

Some of the data pre-processing that is generally involved with machine learning is eliminated with deep learning. These algorithms can handle text and visual data that is unstructured and automate feature extraction, reducing the need for human specialists. Let's imagine, for instance, that we wanted to categories a collection of images of various pets by "cat," "dog," "hamster," etc. Deep learning algorithms can decide which characteristics—like ears—are most crucial for differentiating one species from another. This hierarchy of features is created manually by a human specialist in machine learning. The deep learning algorithm then fine-tunes and adapts itself for accuracy through the processes of gradient descent and backpropagation, enabling it to make predictions about a fresh animal shot with greater accuracy.

Farnaz [5] describes the differences between the convolutional layers and dense neural network layers in his paper. Along with other key contributing factors like materials and window-to-wall ratios, a building's self-shading shape dramatically affects the amount of direct sunlight it receives and contributes significantly to the building's operational energy use. Deep Learning has the potential to help designers and engineers by accurately forecasting the energy efficiency of buildings. In this study, the relevance of two alternative neural network architectures—the dense neural network (DNN) and the convolutional neural network (CNN)—to the prediction of operational energy consumption in buildings in relation to building geometry is evaluated. The performance, simplicity, and computation time comparison between the two neural networks demonstrates that the DNN model outperforms the CNN model. However, the advantage of using architectural visuals in image-based CNN makes design communication easier.

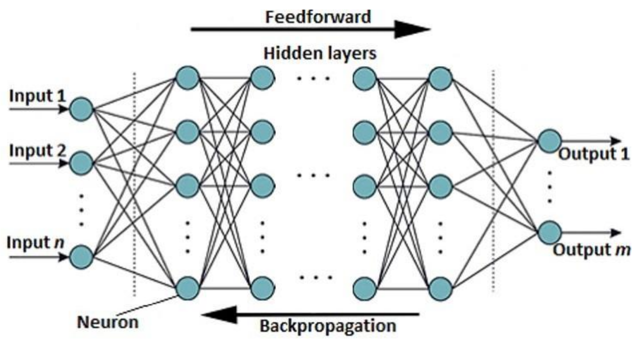


Fig.2. graphically explains the dense neural network with the forward and backward propagation. The Dense Neural Network takes multiple features as the input from the previous layers and makes the classification or predication accordingly.

C. Carla Simulation

CARLA is an open-source simulator for autonomous vehicles. It was created from the ground up to act as a flexible and modular API to handle a variety of jobs related to the autonomous driving issue. One of CARLA's key objectives is to assist in democratizing autonomous driving research and development by acting as a tool that people can quickly use and modify. To do this, the driving simulator must be able to satisfy the demands of various use cases (e.g., learning driving policies, training perception algorithms, etc.). CARLA runs on Unreal Engine and defines highways and urban settings according to the current version (1.4) of the OpenDRIVE standard. An API handled in Python and C++ that is always expanding as the project grows provides control over the simulation. CARLA grew into an ecosystemof projects, created around the primary platform by the community, to facilitate the process of designing, training, and evaluating driving systems. In order to properly appreciate CARLA's potential, it is crucial to realize some aspects of how it operates. The client-server design of the CARLA simulator is scalable. Everything pertaining to the simulation itself is handled by the server, including sensor rendering, physics calculation, updates on the world-state and its actors, and much more. Running the server with a dedicated GPU would be ideal because it aspires for realistic results, especially when working with machine learning. Theclient side is made up of a collection of client modules that manage the world conditions and the logic of the actors in thescene. The CARLA API (in Python or C++), a layer that actsas mediator between the server and the client and is constantly developing to offer new functions, is used to do this. Alexey [6] from Intel Labs describes about Carla as a free and open-source simulator for testing autonomous vehicles. In order to enable the development, instruction, and validation of autonomous urban driving systems, CARLA has been built from the ground up. CARLA offers open digital assets (urban layouts, buildings, and vehicles) in addition to open-source technology and protocols that were developed for this purpose. Flexible selection of sensor suites and ambient variables is supported by the simulation platform. Using CARLA, we compare the effectiveness of three autonomous driving strategies: a conventional modular pipeline, an end-to-end model trained using imitation learning, and an end-to-end model trained using reinforcement learning. The techniques' performance is assessed using metrics provided by CARLA, and the evaluations are conducted in controlled scenarios of increasing difficulty, demonstrating the platform's usefulness for research on autonomous vehicles game play, among other things. Deep Q Networks may be made up of convolutional neural networks and other structures that employ particular techniques to learn about different processes, as opposed to being a specific name for a particular neural network build.

a) Q – Learning: The agent uses an accurate matrix created by the Q-Learning algorithm to optimise its reward over time. This strategy only works in constrained environments with little room for observation because adding more states or actions results in the erroneous algorithm. Behaviour. Based on the Bellman Equation, Q-Learning is a model-free, of-policy RL technique, where v denotes the equation's ideal value.

$$v(s) = E[Rt + 1 + \lambda v(St + 1)|St = s \qquad (1)$$

$$Q^* \ (s, a) \ E's \ [\ r + \lambda \max Q^*(s', a')|s, a] \qquad (2)$$

Where the eq.(2) gives the optimal Q – value. Through iteration policy, which creates a loop between policy evaluation and policy improvement, Q-Learning aims tooptimise the Q-value. As a result of the most recent policy improvement, the greedy policy used in policy assessment assesses the value of function V. The policy is updated, however, with the action that maximises the V function for each state under policy improvement.

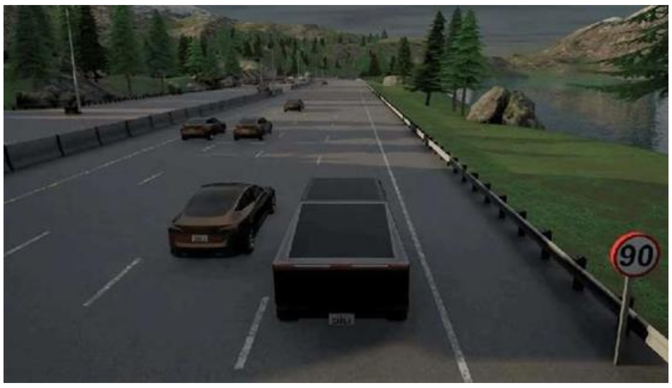


Fig.3. Carla Simulated Environment

$$v(s) = \max E[Rt + 1 + \gamma v(St + 1)|St = s, At = a]$$

$$= \max \textstyle\sum_{a^{F},s} p(s', r|s, a)[r + \gamma v(s')] \quad (3)$$

In fig.3, the simulated environment for Carla is described. The environment can be accessed through python or C++ APIs, and in this study, the agent is connected to the Carla simulation with the aid of a python plugin.

D. OpenAI Gym

The OpenAI gym is a place where learning agents can be created and tested. Although it is concentrated and most appropriate for reinforcement learning agents, it does not prevent one from experimenting with alternative techniques, such as hard-coded game solvers or other deep learning techniques. Greg [7] from

OpenAI purposes research toolset for reinforcement learning is called OpenAI Gym. It features a website where users may discuss their results and evaluate the effectiveness of algorithms, as well as a growing collection of benchmark problems that reveal a common interface.

III. DEEP REINFORCEMENT LEARNING ALGORITHM

Reinforcement learning and deep learning are combined in deep reinforcement learning (DRL). It is better able to learn from unprocessed inputs like raw sensors or photos, enabling end-to-end learning, which expands the applications it can be used for in fields like robotics, video games, NLP, computer vision, healthcare, and other areas. There are multiple algorithms under DRL in which, Deep Q – Learning is used in this research to train the agent.

A. Deep Q – Networks

Neural networks (and/or related technologies) known as Deep Q Networks (DQN) make use of deep Q learning to create models that can be used to simulate intelligent video The eq.3. can be further updated to rise the give the normal value based on the state and action.

$$Q(St, at) = Q(St, at) + \gamma \max Q(St + 1, at) - Q(St, at)] \quad (4)$$

Eq.(4) shows the end equation for the Q – Learning model.

b) Deep Q – Learning: When the space of observation expands, Q-learning becomes less general. Imagine a case in which there are 10 states and 10 alternative actions; this is a 10x10 matrix. However, if there are 1000 states, the Q-matrix dramatically grows and becomes tough to manage manually. Deep Q-Learning uses a neural network to handle the two- dimensional array in order to address this problem. DQN uses a learning process to estimate Q-values, with the state serving as the input and the matching Q-value for each action serving as the output. The goal equation y is where the difference between D-Learning and Deep Q-Learning lies.

$$Yj = Rj + \gamma \max Q(Sj + 1 , a'; \theta-) \quad (5)$$

Where the θ stands for the parameters in the Neural Network.

The target Q-network and the replay memory are just two of the crucial heuristics that DQN employs. Rewards are clipped between -1 and +1 to maintain the target values in a realistic range and to guarantee proper learning in practice. Clipping the rewards makes it simpler to apply the same learning rate across several games and reduces the size of the error derivatives (however, it introduces a bias). One method for making an agent avoid terminal states in games were theplayer has multiple lives is to also link the loss of a life to the occurrence of a terminal state (when the discount factor is set to 0).

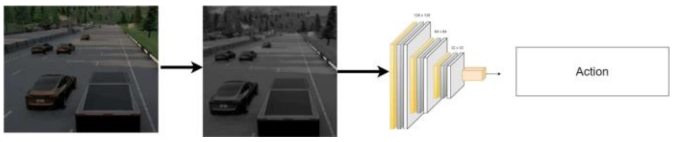


Fig. 4. Internal Computational Block

Numerous deep learning-specific techniques are also employed in DQN. In particular, a preprocessing phase of the inputs is employed to deal with several task-specific issues, minimize input dimensionality, and normalize inputs (it scales pixel values into [-1,1]). Additionally, the neural network function approximator's initial layers are constructed using convolutional layers, and optimization is carried out using the RMSprop variant of stochastic gradient descent.

IV. METHODOLOGY

The research follows a particular methodology which used to design the agent and connect with the Carla environment. The methodology carries various parts which is graphically described in fig. 5.

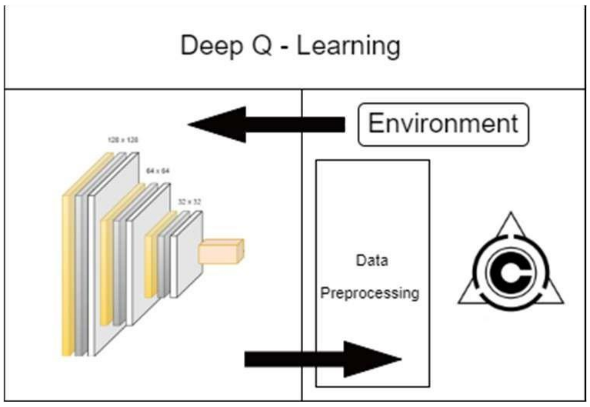


Fig.5. Parts of Agent Design

A. Environment – Agent Interaction

Carla, an artificially intelligent agent, interacts with the environment through a network, where its serves as a server and is constantly alert for commands coming from the client side. The Python APIs or plugins are used by the DQL Agent, a client, to control the environment. The environment has the ability to modify and regulate each and every parameter.

The environment also provides various virtual sensors particularly Camera, LIDAR and Segmentation sensors which is used in this project to monitor the environment. The observation from the environment is collected and further used for the action computation using the internal computational blocks mentioned in the fig.4. The OpenAI gym provides a wrapper which covers the Carla environment, the wrapper helps in connecting with the DQL model in a better way.

Gym consider of various functions blocks such reset, step and render. The reset function will clear the environment an reinitialize the operations and pre – declared variables in the environment. For each episode, the resetting is done by the agent, the reset function will return the current observation of the environment which helps verifying the proper deployment of DQL Agent in the environment.

The Step function, which is responsible for most of the actions in the environment. The step function is initialized by the DQL agent. The function returns the parameters such as observation, state, rewards and information in end of every step. The step is expected to end at every end of every episode. Based on the rewards from the episode, the agent will consider the step in the real=time deployment.

a) Image Pre -processing: The reset and step functions sends an obeservation as a return which is further processed by a convolutional neural network (CNN) and before the image is feed into the CNN layers, the pre – processing is happens with the image. Pre – processing involves steps such a resize of the image to a particular size and converting the RGB image into Black and White format for easy computation.

b) Neural Network:The nerual Network architecture used in the project was resnet which is consider as the one of the best pre trained model. In which, the pre – processed images from the reset and step functions are used to predict the action, the agent should carried out next in the environment.

An artificial neural network called a residual neural network (ResNet) (ANN). It is a gateless or open-gated variation of the HighwayNet, which was the first functionally complete, extremely deep feedforward neural network with hundreds of layers—much deeper than earlier neural networks. To skip some layers, use shortcuts or skip connections (HighwayNets may also learn the skip weights themselves through an additional weight matrix for their gates). Typical ResNet models are constructed with batch normalization in between double- or triple-layer skips that contain ReLU nonlinearities. DenseNets are models that have numerous parallel skips. A non-residual network is referred to as a plain network while discussing residual neural networks. The forward propagation of the ResNet can be give by the eq. 6.

$$a^{l} \coloneqq g(W^{l-1,l}.\, a^{l-1} + b^{l} + W^{l-2l}.\, a^{l-2}) \\ \coloneqq g(z^{l} + W^{l-2l,l}.\, a^{l-2}) \quad (6)$$

Where a^{l} denotes the activation function which helps in producing non – linearity in the network. Further the equation can be updated as.

$$a^{l} \coloneqq g(Z^{l} + \sum_{k=2}^{k} W^{l-2,l}.\, a^{l-k}) \quad (7)$$

V. RESULTS AND DISCUSSION

The research mainly focused on designing an artificial intelligent agent using Deep Q – Learning (DQL) which can be trained by the Carla, as mentioned before is an open – source simulator for training and testing the AI agents. The main matrices focused in understanding the performance of the AI Agent is the maximum rewards per episode. The higher rewards which in turns associated with the better action which had made during the training stage.

A. Memory Management

The Deep Q – Learning algorithm requires better memory management method and high intensity of computation for training the artificial intelligence agent. The training was carried on top of the NVIDIA's RTX 3040 GPU. The memory was handled by allocating the memory in the GPU dynamically.

B. Training the Agent

The training of the agent as mentioned requires a high intensity of computation and effective memory handling. The policy which is used in the compilation of agent called" ϵ – greedy policy". By randomly selecting between exploration and exploitation, Epsilon-Greedy is a straightforward strategy for balancing discovery and exploitation. Epsilon refers to the probability of deciding to explore, and the epsilon-greedy, with a low probability of exploring, exploits most of the time.

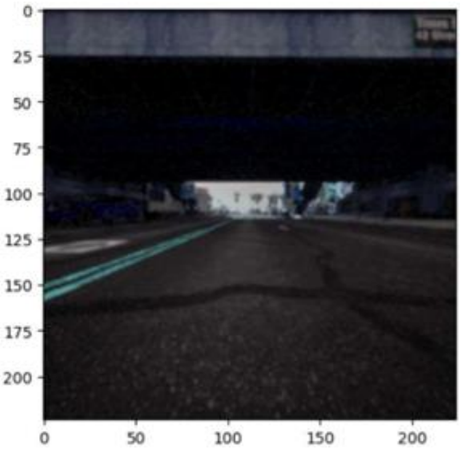


Fig. 6. Sample Image from Reset Return

Functions such Reset and Step plays an important role in training the AI agent. For each episode, the environment is reset and the step is made by the agent to move from one sub – state to goal or next sub- state.

Fig.7. shows the graph between episodes and rewards grained. The model trained good at some point and grained the maximum reward of +70 and the negative rewards due to possible collusion with the objects in the environment is about -10. The steps are fixed constant which is no. of steps = 2000 and the GPU utilization is

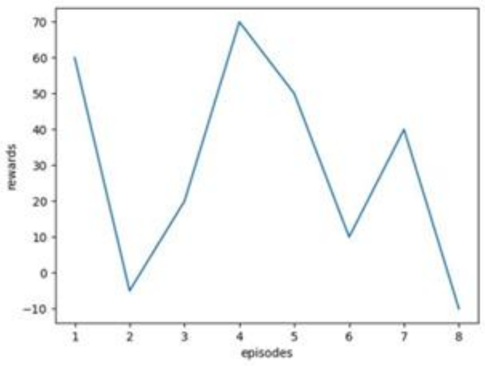


Fig.7. Episode Vs Reward Graph

VI. CONCLUSION

Hence the research concludes by suggesting the framework OpenAI gym is really helpful in creating the customized environment for the training for AI agent. Dynamic memory allocations and improved policies can help in improving the training accuracy of the agent. The parameters such Higher end GPU and increasing steps with longer episodes can reduce the negative rewards in training the agent.

VII. REFERENCES

[1]. Perez-Gil, Óscar, et al. "Deep reinforcement learning based control for autonomous vehicles in carla." Multimedia Tools and Applications 81.3 (2022): 3553-3576.

[2]. Syavasya, C. V. S. R., and A. Lakshmi Muddana. "Optimization of autonomous vehicle speed control mechanisms using hybrid DDPG- SHAP-DRL-stochastic algorithm." Advances in Engineering Software 173 (2022): 103245.

[3]. Liu, Teng, et al. "Decision-making at unsignalized intersection for autonomous vehicles: Left-turn maneuver with deep reinforcement learning." arXiv preprint arXiv:2008.06595 (2020).

[4]. Botvinick, Matthew, et al. "Reinforcement learning, fast and slow." Trends in cognitive sciences 23.5 (2019): 408-422.

[5]. Nazari, Farnaz, and Wei Yan. "Convolutional versus Dense Neural Networks: Comparing the Two Neural Networks Performance in Predicting Building Operational Energy Use Based on the Building Shape." arXiv preprint arXiv:2108.12929 (2021).

[6]. Dosovitskiy, Alexey, et al. "CARLA: An open urban driving simulator." Conference on robot learning. PMLR, 2017.

[7]. Brockman, Greg, et al. "Openai gym." arXiv preprint arXiv:1606.01540 (2016).

Cite this article as :

Bhavanisankari. S, Vimal William, Sachin. A, "Design and Training of AI Agent using Deep Q - Learning and Carla", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 10 Issue 1, pp. 74-81, January-February 2023. Journal URL : https://ijsrset.com/IJSRSET2310110