IJCRT.ORG ISSN : 2320-2882
INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

GAME PLAYING AI USING REINFORCEMENT LEARNING

Deepak Gupta
Department of InformationTechnology
Shree L.R. Tiwari College of Engineering
Mumbai, India

Neeladri Chatterjee
Department of InformationTechnology
Shree L.R. Tiwari College of Engineering
Mumbai, India

Harsh Dubey
Department of InformationTechnology
Shree L.R. Tiwari College of Engineering
Mumbai, India

Komal A. Champanerkar
Assistant Professor
Department of InformationTechnology
Shree L.R. Tiwari College of Engineering
Mumbai, India

***Abstract*— Reinforcement Learning is a category of machine learning, that is based on evaluative feedback. It is about taking suitable actions and maximising rewards, which is based on feedback mechanisms. Reinforcement Learning has proven to be the state of arts method for a lot of tasks related to Artificial Intelligence. This paper presents implementation of one of the most important reinforcement learning algorithms. The project uses 3 games in which the agent will learn control policies using reinforcement learning approaches to achieve a high score. The project explores two deep reinforcement learning approaches, Markov Decision Process and Deep Q-Learning, both of them are proposed by DeepMind to train intelligent agents that can interact with an environment with the help of automatic feature engineering and thus requiring minimal domain knowledge.**

Keywords*— *Reinforcement Learning, Q-Learning, Markov Decision Process.

I. INTRODUCTION

As there is tremendous growth in the field of Artificial Intelligence and multiple research is going on, there is a need for a technique or method where the system or model can itself learn to perform tasks like we humans do it. Reinforcement learning is such an algorithm that works in a similar way to humans. Reinforcement Learning (RL) is an area of machine learning that operates via an idea of reward or punishment for every action an agent takes and the end goal for the agent is to maximize the cumulative reward. We are going to be discussing Q-Learning and Deep Q Network in this paper. However, these are not the only RL algorithms that exist. MDP (Markov Decision Process), SARSA, and DDPG are some of the Reinforcement Learning algorithms that are most popular today. We are going to implement these Reinforcement Learning Concepts in Games. Games provide interesting and complex problems/scenarios for agents to solve which makes video games perfect environments for research in Artificial Intelligence. These virtual environments are safe and controllable. In addition, these game environments provide an infinite supply of useful data for machine learning and AI algorithms, and they are much faster than real-time. These characteristics make games the unique and preferred domain for AI research. On the other side, Artificial Intelligence has been helping games to become better in the way we play, understand and design them.

There are also some problems or challenges like learning proper policies to make decisions in a dynamic unknown environment is difficult. For this problem, data driven methods such as supervised learning and reinforcement learning (RL) are feasible solutions. In addition the challenge is that most game AI is developed in a specified virtual environment. Transfering AI models ability among different games is a core challenge. A more general learning system is also necessary.

I.I. SIGNIFICANCE AND BACKGROUND

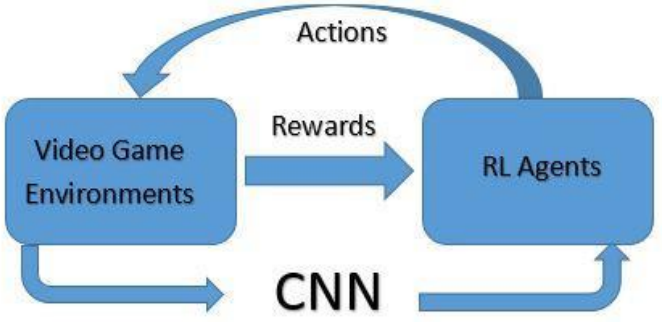


Fig. 1. Typical DRL for video games

How is RL Algorithms different from machine learning algorithms? Let's take a step back and think about supervised learning and unsupervised learning. In those kinds of situations you have a pre-existing set of inputs and finite outputs corresponding to the input, or a cluster that you want the input to fall in. So you just train the data you collect in one of the supervised or unsupervised algorithms you see fit.However, reinforcement learning works in a completely different way. They can deal with large complex problem spaces, meaning it can deal with every possible combination of possible circumstances. RL learns by the process of obtaining rewards or punishments for every action it performs, and is able to adapt in unforeseen circumstances accordingly. This can not only be applied to tasks like game playing, where the set of actions and outcomes are virtually infinite but also tasks like Natural Language Processing and Supply Chain Management
can take advantage of RL.
Generally, training an agent to make decisions using high-dimensional inputs is difficult.With the development
of deep learning and AI, researchers take neural networks as function approximation, and use plenty of samples to optimize

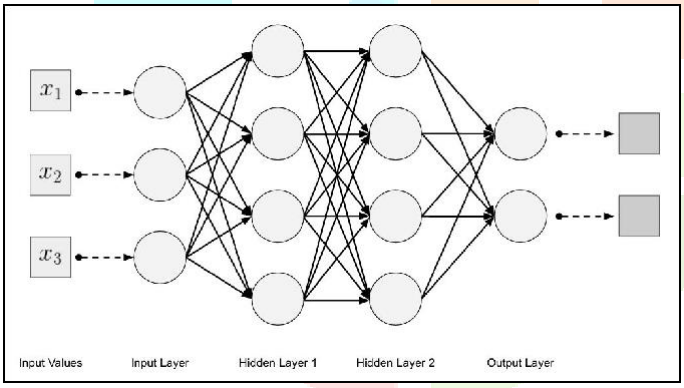
policies. The framework diagram of typical DRL for video games is depicted in above Fig. 1.

II. CONCEPTS

A. DEEP LEARNING

Deep Learning is an Artificial Neural Networks, and is used to learn data representation. It is inspired by the theory of brain development. It can be used in all three supervised, unsupervised and semi-supervised learning. In this project we are using one of the neural networks of deep learning. Some of the classes of Deep Learning are CNN, RNN etc. Convolutional Neural Network (CNN) is widely applied in computer vision. It is inspired by biological processes, and is shift invariant based on shared-weights architecture. Recurrent Neural Network is another kind of deep neural network especially for NLP-Natural Language Processing. Deep learning architectures have been applied into many fields, and have achieved significant successes, such as speech recognition, image classification and segmentation, semantic comprehension, and machine translation. DL-based methods with efficient parallel distributed computing resources can break the limit of traditional machine learning methods. This method inspires scientists and researchers to achieve more and more state-of-the- art performance in respective fields.

B. DEEP NEURAL NETWORK



- convolution, pooling, non-linearity
- multi-layer neural network (MLP) as final classifier
- sparse connection matrix between layers to avoid large computational cost
- dropout technique to selectively ignore single neurons during training, a way to avoid overfitting of the model.

C. DEEP REINFORCEMENT LEARNING

DRL makes a combination of DL and RL, achieving rapid developments since proposed. This section will introduce various DRL methods, including value-based methods, policy gradient methods, and model-based methods.

i. Value-based DRL

Deep Q-network (DQN) is the most famous DRL model which learns policies directly from high-dimensional inputs. It receives raw pixels, and outputs a value function to estimate future rewards. DQN uses theexperience replay method to break the sample correlation, and stabilizes the learning process with a target Q-network.
The Loss function equation is given by.

$$L_i(\theta_i) = E_{(s,a,r,s')\sim U(D)}[(y_i^{DQN} - Q(s,a;\theta_i))^2]$$

and

$$y_i^{DQN} = r + \gamma \max_{a'} Q(s',a';\theta_i^-).$$

DQN introduces double Q Learning to reduce observed overestimations, and it leads to much better performance. Prioritized experience replay helps prioritize experience to replay important transitions more frequently.

ii Model Based DRL

Combining model-free reinforcement learning with on-line planning is a promising approach to solve the sample efficiency problem. TreeQN is proposed to address these challenges. It is a differentiable, recursive, tree-structured model that serves as a drop-in replacement for any value function network in DRL with discrete actions.

iii Policy Based DRL

Policy gradient DRL optimizes the parameterized policy directly. Actor-critic architecture computes the policy gradient using a value-based critic function to estimate expected future reward. Asynchronous DRL is an efficient framework for DRL that uses asynchronous gradient descent to optimize the policy. Asynchronous advantage actor-critic trains several agents on multiple environments, showing a stabilizing effect on training. The objective function of the actor is demonstrated as

$$J(\theta) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} A_{\theta,\theta_v}(s_t,a_t)\log\pi_\theta(a_t|s_t) + \beta H_\theta(\pi(s_t))]$$

D. MULTIAGENT LEARNING

Multi-agent learning is very important in video games, such as StarCraft. In a cooperative multi-agent setting, curse of-dimensionality, communication, and credit assignment are major challenges.Team learning uses a single learner to learn joint solutions in multi-agent systems, while concurrent learning uses multiple learners for each agent. Recently, the centralised training of decentralised policies is becoming a standard paradigm for multi-agent training. Multi-agent DDPG considers other agents' action policy and can successfully learn complex multi-agent coordination behavior. Counterfactual multiagent policy gradients use a centralized critic to estimate the action-value function and decentralized actors to optimize each agents' policies, with a counterfactual advantage function to address the multi-agent credit assignment problem. In addition, communication protocols are important to share information to solve multi-agent tasks.

III. METHOD / IMPLEMENTATION

In this section we are going to be discussing how we created an environment and how different techniques were used to implement the model and to train our agents.

Q-Learning

Q-Learning is a model-free form of machine learning, in the sense that the AI "agent" does not need to know or have a model of the environment that it will be in. The same algorithm can be used across a variety of environments. First, we explore Q-Learning to create an agent that can play Snake Game and Flappy Bird. Q-Learning is a foundation for Deep Q-Learning.

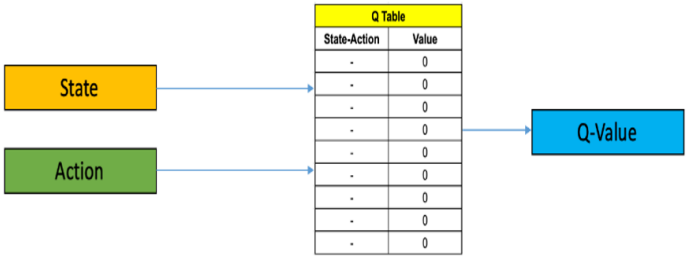
$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

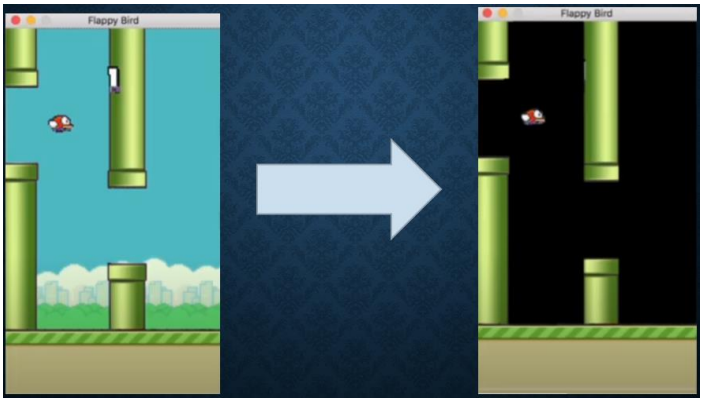
The above equation is known as Bellman Equation which is used to calculate a new Q value each time the agent loops through a state. Q-Learning is an algorithm that improves the agent based on the reward that the agent gets for performing an action. A major part of Q-Learning is the Q table which consists of Q values for each set of actions or movements an agent can take. "Learning" in this case is updating the Q table so that we have the optimal Q value for every action in each state. Once learning is done, the agent can refer back to the Q table to figure out the best possible action for any given state. In this
paper, we use Q-Learning to train an agent to play the Snake Game and flappy bird. Most of the AI implementations use hardcoded search algorithms. However, we don't have perfect knowledge of the game in a lot of cases and that is where search algorithms fail and reinforcement learning comes in. One of the major issues with Q-Learning is memory and time. The amount of memory required to save Q values for every possible action in a fairly big game is huge and the time required to explore each state to create the required Q table is simply unrealistic. This is where Deep Q Networks come in. Deep Q-Learning is simply a way of avoiding the Q table as a whole and rather use a neural network to approximate the Q value given the state. In this paper, we use Deep Q-Learning to train our agents to play more complicated games.

In Q-Learning, we build a Q table for all possible sets of states and actions. But we can sometimes encounter states which might be similar, but not exactly the same as one of the combinations in the Q-Table. This is where Deep Q-Learning comes in. Flappy bird has a relatively small state space so it is easy to discretize the state space and create a Q-table for it. But once we move to more complex games, the state space and action space are way bigger than that
of flappy birds and hence the size of the Q-table increases exponentially. Storing such a Q table would be next to impossible.

ALGORITHM

INITIALIZE REPLAY MEMORY
INITIALIZE DQN TO RANDOM WEIGHTS
REPEAT
  NEW EPISODE (NEW GAME)
  INTIALIZE STATE S_0
  REPEAT
    EXTRACT XT FROM RAW PIXEL DATA UPDATE STATE S_T WITH X_T
    ADD EXPERIENCE
    E_T= ($\phi(S_t-1)$, A_t-1, R_t-1, $\phi(S_t)$) TO REPLAY MEMORY
    TAKE BEST ACTION
    A_T=ARG $MIN_{a \in actions}$Q (ST, A) WITH EXPLORATION
    IF TRAINING
  UNIFORMLY SAMPLE A BATCH OF EXPERIENCES FROM THE
  REPLAY MEMORY
  BACKPROPAGATE AND UPDATE DQN WITH THE MINIBATCH
  UPDATE EXPLORATION PROBABILITY E
  IF C UPDATES TO DQN SINCE LAST UPDATE TO TARGET
  NETWORK THEN
    UPDATE THE TARGET Q-NETWORK
    Q(S,A) ← Q(S,A)
  END
  UPDATE STATE S_T WITH A_T
  UPDATE CURRENT REWARD R_T AND TOTAL REWARD
  UPDATE GAME PARAMETERS (BIRD POSITION, SNAKE GRID ETC.)
  REFRESH
    UNTIL FLAPPY BIRD CRASHES OR SNAKE COLLIDES;
  RESTART GAME
UNTIL CONVERGENCE OR NUMBER OF ITERATIONS REACHED;

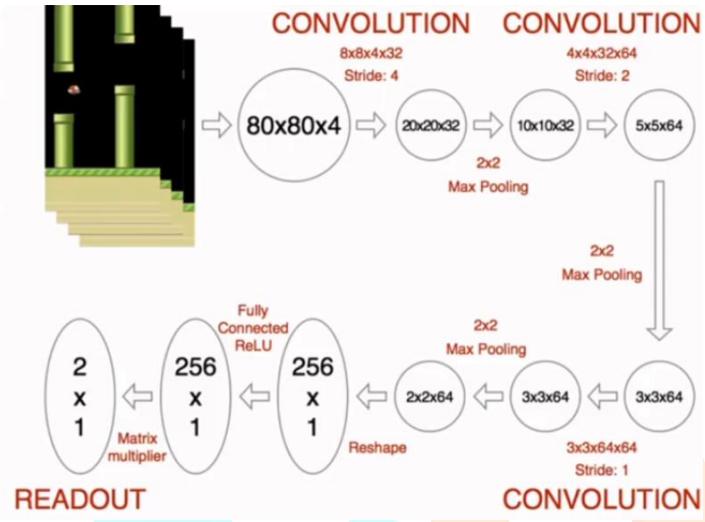




Removal of Pixels

Our Deep Q-Network is trained on raw pixel values observed from the game screen at each time step. We feed the raw images to a Convolutional Neural Network with Max Pooling layers and the output layer has the same dimension as the action space, which in this case is two. One of them corresponds to moving the bird upwards and the other one corresponds to doing nothing. At each time step, the network performs whichever action corresponds to the highest Q-value using a greedy policy. The network is trained using the algorithm shown above.

Given Below is the Overview Diagram of the Working of the Implementation



III. APPLICATIONS

Application in Self Driving Car- Deep Reinforcement Learning is used for autonomous driving. In self-driving cars there are various aspects which are to be considered such as speed limits at various places, drivable zones, avoiding collisions etc. just to mention a few. Some of the autonomous driving tasks where reinforcement learning could be applied include trajectory optimization, motion planning, dynamic pathing, controller optimization, and scenario based learning policies for highways. Parking Policies can be also used which will help in parking the vehicle.

Applications in HealthCare- In healthcare, patients can receive treatment from policies learned from RL systems. RL is able to find optimal policies using previous experiences without the need for previous information on the mathematical model of biological systems. It makes this approach more applicable than other control-based systems in healthcare

Application in Software Engineering- Facebook has developed an open-source reinforcement learning platform Horizon. The platform uses reinforcement learning to optimize large-scale production systems. Facebook has used Horizon internally to personalize suggestions, deliver more meaningful notifications to users ,optimize video streaming quality.

Some other Applications where RL is used are in Marketing, News Recommendation , Robotics etc.

IV. ANALYSIS

Reinforcement Learning has a lot of potential. Game playing is one small thing that RL is good at and it has been explored extensively in recent years. Recently, a new toolkit called NLPGym has been released which gives the user an environment where they can test Reinforcement Learning on Natural Language Processing tasks.Our Team would love to explore this, which I really look forward to. Future work building on this project might include accomplishing tasks that have a bigger action and state spaces and possibly replicating some of more complicated papers like Grandmaster level in StarCraft II using multi-agent reinforcement learning by Vinyals etc.. Future of reinforcement learning is very bright and I can see these techniques being applied in a lot of fields.

ACKNOWLEDGMENT

Gratitude might be a way to express one's thankfulness but in a view it is still less in showing how invaluable the guidance and the help means. My obedient and sincere thanks to Dr. S. Ram Reddy, Principle, SLRTCE, Mira Road for providing the facilities and necessary assistance to our requirements which was crucial for the completion of work. I express my thanks to Prof. Sunil Yadav, HOD(IT), SLRTCE, Mira Road, for giving his support for the growth of the project work.

I am very grateful to Prof. Komal Champanerkar., Project Supervisor under whose supervision it was made possible to lead the project down its correct path. I also thank her for being our Technical Guide throughout for the project whose presence, guidance, help and encouragement towards the project included an effectiveness in the improvement of the quality of the project at all stages. Their guidance and help will surely serve as the source and inspiration in the future, forever an invaluable knowledge.

I am also extremely thankful to the authors whose previous works have been a great help in the consultation and reference for the project. I also thank my friends. I appreciate their encouragement and help which has been a cause for my positivity towards the outcome.

I convey my gratitude to my family whose support and blessing were the backbone in making the project come true.

All the thanks and my gratitude towards my teachers, family, friends, the efforts, the blessings and the guidance which had made it possible to achieve the fulfilment of the assignment.

REFERENCES

[1] Vinyals, O., & Babuschkin, I. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning.

[2] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2015). Playing Atari with Deep Reinforcement Learning

[3] I. P. Pinto and L. R. Coutinho, "Hierarchical reinforcement learning with monte carlo tree search in computer fighting game," IEEE Transactions on Games, vol. 11.3, pp. 290–295, 2018

[4] Y. Li, "Deep Reinforcement Learning: An Overview," CoRR, abs/1701.07274, last revised Nov. 26, 2018

[5] S. Tom, Q. John, A. Ioannis, and S. David, "Prioritized experience replay," in International Conference on Learning Representations, 2016

[6] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. MIT Press, 1998.