ISSN: 2584-0495 Vol. 2, Issue 9, pp. 1181-1186

International Journal of Microsystems and IoT

ISSN: (Online) Journal homepage: https://www.ijmit.org

AI Game Playing: Using Deep Reinforcement Learning

M. Navdeep, B. Ankayarkanni, J. Albert Mayan, M.D. Anto Praveena

Cite as: Navdeep, M., Ankayarkanni, B., Mayan, J. A., & Praveena, M. D. A. (2024). AI Game Playing: Using Deep Reinforcement Learning. International Journal of Microsystems and IoT, 2(9), 1181-1186. https://doi.org/10.5281/zenodo.14098981

Published online: 23 Sept 2024

DOI: https://doi.org/10.5281/zenodo.14098981

International Journal of Microsystems and IoT
Vol. 2, Issue 9, pp.1181-1186; DOI: https://doi.org/10.5281/zenodo.14098981

AI Game Playing:Using Deep Reinforcement Learning

M. Navdeep, B. Ankayarkanni, J. Albert Mayan, M.D. Anto Praveena

Department of Computer Science and Engineering, Sathyabama Institute of Science and Technology, Chennai. India

ABSTRACT

In the evolving landscape of artificial intelligence, Deep Reinforcement Learning (DRL) has carved a niche as a transformative methodology for training agents in complex and dynamic tasks. This study ventures into the realm of video game AI, specifically targeting the renowned and challenging Mario game series. Our research is anchored on deploying and refining cutting-edge DRL techniques, including Deep Q-Networks (DQNs), Advantage Actor-Critic (A3C), and Proximal Policy Optimization (PPO). The primary objective is to cultivate an AI agent with the acumen to adeptly traverse and excel in various levels of Mario games, with the aspiration of achieving and potentially surpassing human-level performance. By leveraging the multifaceted nature and popularity of Mario games, this research contributes to the understanding and advancement of DRL in navigating environments that closely resemble real-world complexities and decision-making scenarios.

KEYWORDS

Advantage Actor-Critic
AI Game Playing
Deep Reinforcement Learning,
Deep Q-Networks,
Proximal Policy Optimization,

1. INTRODUCTION

Video games, with their intricate virtual landscapes and challenging mechanics, have become a fertile ground for advancing AI techniques. Deep Reinforcement Learning (DRL) has risen as a formidable approach in this realm, transforming how AI systems learn and master complex interactions within their environments. This research is centered on applying DRL to video gaming, specifically focusing on the iconic Mario series. The Mario games, with their diverse challenges and dynamic environments, provide an ideal platform for testing and enhancing AI capabilities. We aim to leverage DRL to equip an AI agent with the skills to excel in Mario games, potentially exceeding human expertise. Our choice of Mario games as a testbed is intentional, considering their mix of strategic decision-making, timing precision, and evolving challenges that mirror real-world complexities.

The choice of Mario games as our testbed is deliberate. These games offer an amalgamation of obstacles, enemies, and ever-evolving environments that simulate world challenges. Navigating Mario through these digital worlds requires not only precise timing but also strategic decision-making. Furthermore, the platform offers a dynamic training ground to assess an AI agent's adaptability and learning capability in the face of a plethora of challenges.

In this paper, we detail the methodology employed in applying DRL algorithms, including Deep Q-Networks (DQNs), Advantage Actor-Critic (A3C), and Proximal Policy Optimization (PPO), to the task of training an intelligent Mario-playing agent. We elucidate the intricacies of our experiments, presenting empirical evidence of the agent's performance in terms of win rates, scores, and its capacity to surmount the myriad complexities presented by various Mario game variants.

The Mario game environment, a pivotal component of the iconic video game series, serves as the immersive backdrop for Mario's adventures. Designed by game developers, these digital worlds are characterized by their diversity, featuring distinct themes, challenging obstacles, and interactive elements. Players navigate through a dynamic landscape of hazards, platforms, and enemies, all while collecting power-ups like Super Mushrooms and Fire Flowers to enhance Mario's abilities.

The environments offer players a rich tapestry of experiences, with secrets to uncover, coins to collect, and end-of-level goals to achieve, often culminating in epic boss battles. Over the years, the Mario game environment has evolved and adapted, offering a compelling blend of entertainment and challenge. For researchers and developers, the Mario game environment provides a captivating and well-defined space for experiments in artificial intelligence, particularly in the context of deep reinforcement learning. This dynamic digital realm continues to inspire exploration and innovation in the field of gaming and AI.

2. LITERATURE SURVEY

Deep Reinforcement Learning (DRL) has emerged as a potent tool in the development of intelligent agents capable of mastering complex tasks, including playing video games at or above human performance levels. The seminal work by Mnih et al. (2015), introducing the Deep Q-Network (DQN), marked a significant advancement in the field, demonstrating the ability of DRL agents to learn optimal policies directly from high-dimensional sensory inputs through end-to-end learning. DQN combines Q-learning with deep neural networks, enabling the agent to handle the challenges of partial observability and high-dimensional state spaces inherent in video games [1].

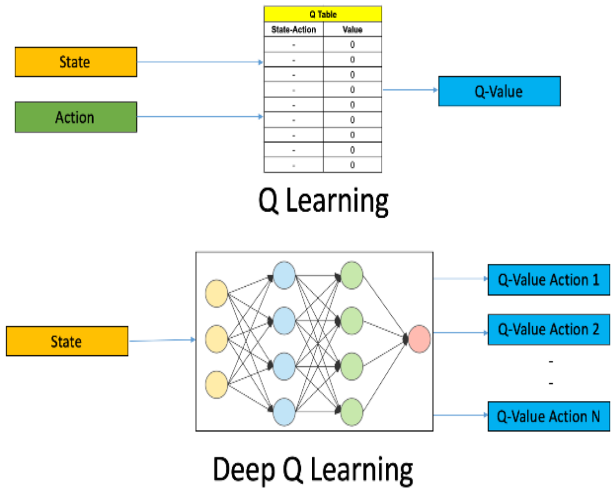

Fig. 1 Proposed Algorithm DQN

Further developments in DRL have aimed at addressing the limitations of DQN, such as overestimation of action values and difficulty in learning from scarce rewards.[2] The introduction of Double Deep Q-Networks (DDQN) by Van Hasselt et al. (2016) proposed an elegant solution to the overestimation problem by decoupling the selection and evaluation of actions. This modification significantly improved the stability and performance of DRL agents across a range of environments.

The application of DRL in video games serves not only as a benchmark for the capabilities of AI agents but also offers insights into the generalization, scalability, and robustness of these learning algorithms. Tackling games like Super Mario Bros, with their rich environments and complex decision-making requirements, presents a unique challenge that has driven research towards more sophisticated models and training techniques.

2.1 Used Model

The model used in this research is based on the Double Deep Q-Network (DDQN) architecture, which enhances the standard DQN by addressing the overestimation bias observed in the original algorithm. The DDQN consists of two key components: an online network for selecting actions and a target network for evaluating those actions. This separation helps in stabilizing the learning process and improving the accuracy of Q-value estimation [5].

The network architecture comprises several convolutional layers followed by fully connected layers. The convolutional layers are responsible for extracting features from the input state representation, which in the context of Super Mario Bros, is raw pixel data from the game screen. The fully connected layers map these features to a set of possible actions, each associated with a Q-value indicating the expected return of taking that action in the given state.

Training of the DDQN model involves updating the online network using gradients from the temporal difference error between the predicted Q-values and the target Q-values, which are periodically copied from the online network to the target network. This process is repeated over episodes of gameplay, with the agent's policy gradually converging to an optimal strategy through exploration and exploitation[6].

In conclusion, the DDQN model represents a significant step forward in the application of DRL to video games, offering a robust framework for training AI agents to navigate complex environments and make strategic decisions based on learned experiences.

3. MARIO ENVIRONMENT

[7] Super Mario Bros presents a multifaceted challenge for deep reinforcement learning (DRL) models. The dynamic elements within the game, such as Goombas, Koopa Troopas, and Piranha Plants, inject an element of unpredictability, demanding real-time adaptability from the DRL agent.[8] Navigating through the pixelated landscapes requires the model to decipher and respond to the intricate dance of movement patterns exhibited by in-game entities, pushing the boundaries of the model's learning capabilities. Concurrently, the acquisition and utilization of power-ups and items become pivotal aspects of decision-making. Super Mushrooms, Fire Flowers, and Stars not only serve as tools for enhanced abilities but also act as catalysts for the agent's strategic adjustments. [3] The DRL model dynamically tailors its behavior based on the availability and deployment of these items, introducing a layer of complexity that goes beyond conventional pathfinding challenges.

Fig.2 Mario Environment

Level design in Super Mario Bros. further amplifies the testing ground for DRL algorithms. Each level unfolds as a unique amalgamation of platforms, enemies, and obstacles, demanding adaptability from the agent to traverse varying degrees of difficulty. The diversity in level configurations underscores the robustness and generalization capabilities of the DRL model, showcasing its aptitude for analyzing and responding to an array of environmental complexities. Hidden elements, such as warp zones and secret areas, add an exploration dimension to the game, necessitating the model's discovery and utilization of shortcuts. This dynamic interplay of environmental intricacies underscores the comprehensive nature of the challenges within the Mushroom Kingdom and demonstrates the DRL model's ability to evolve its strategies amidst a plethora of changing conditions.

4. PROCESSING TECHNIQUES

A critical aspect of training an AI agent in a complex environment like Super Mario Bros is the efficient representation of the game's state. To facilitate this, we employed several preprocessing techniques aimed at reducing the complexity and size of the state space, thereby enhancing the learning efficiency of our deep reinforcement learning (DRL) model.

The first preprocessing step involves implementing a frame skipping strategy. By integrating a Skip Frame wrapper, we modified the environment to return only every fourth frame. This approach effectively reduces the frequency at which the agent receives state updates, thereby simplifying the decision-making process and expediting the training procedure. The wrapper accumulates rewards over the skipped frames and terminates early if a terminal state is reached, ensuring that critical information is not overlooked [4].

The implementation of Skip Frame is a testament to the balance between maintaining game dynamics and computational efficiency. The second preprocessing technique is the conversion of RGB images to grayscale. Using the Gray Scale Observation wrapper, we transformed the original RGB states into grayscale. This change significantly reduces the dimensionality of the input data, allowing the neural network to process the states more efficiently without substantial information loss. The conversion to grayscale lowers the computational burden and focuses the model's attention on the structural elements of the game state, which are pivotal for gameplay.

5. TRAINING AND EVALUATION

[9] The training and evaluation of the Double Deep Q-Network (DDQN) model for the Super Mario Bros game are pivotal steps in our research. These processes ensure that the AI agent not only learns effectively but also generalizes well across various game scenarios.

The training of the DDQN model is conducted over several episodes, each representing a complete game of Super Mario Bros from start to finish or until the agent loses. The key aspects of the training procedure include

Initialization: The environment is initialized, and the Mario agent is instantiated with the state and action dimensions and the computational capabilities.

Experience Gathering: During each episode, the agent interacts with the environment by choosing actions based on the current policy. The states, actions, rewards, and subsequent states are stored in a replay buffer.

Learning from Experience: At regular intervals, the Mario agent samples a batch of experiences from the replay buffer and uses them to update the online network. This step involves calculating the TD estimate and TD target, computing the loss, and performing backpropagation.

Network Synchronization: Periodically, the weights of the target network are synchronized with the online network, ensuring the stability of the target values used in the TD target calculation.

Logging and Monitoring: Key metrics such as rewards, losses, and Q-values are logged after each step and episode. This data is crucial for monitoring the agent's performance and learning progress.

[10] Tampuu et al. (2017) explores the use of DRL in multiagent settings, both cooperative and competitive, which is highly relevant to complex game environments. The results of the training are analyzed in terms of the improvement in game performance over episodes. We track the evolution of the agent's policy, observing how it adapts its strategy to different challenges within the game. The learning curves, showing the progression of rewards and losses, provide insights into the learning efficiency and stability of the DDQN model. [11] This paper extends the DQN framework to handle partially observable environments using recurrent neural networks.

Throughout the training process, we encountered challenges such as balancing exploration and exploitation, tuning hyperparameters for optimal performance, and ensuring sufficient diversity in the experience replay buffer. Solutions included adjusting the exploration rate over time, conducting hyperparameter sensitivity analysis, and implementing strategies to ensure a varied sampling of experiences.

In conclusion, the training and evaluation of the DDQN model in the Super Mario Bros environment are critical for developing a robust and capable AI agent. Our methodology demonstrates the effectiveness of DDQN in mastering complex video games and highlights the importance of systematic training and thorough evaluation in advancing the field of game-playing AI.

6. RESULTS

This section of the paper presents a detailed analysis of the results obtained from the training of the Double Deep Q-Network (DDQN) model in the Super Mario Bros environment. We focus on the performance metrics of the AI agent, its learning progress over episodes, and significant observations made during the training.

The AI agent was trained over several episodes, each representing a complete game session of Super Mario Bros. The key performance metrics observed during training include:
The table displays a comprehensive overview of the AI agent's performance metrics across multiple training episodes. Each row in the table corresponds to a specific episode, with the episodes listed in ascending order. The columns provide detailed data on various aspects of the agent's performance during each episode. The contents of each column are as follows:

Episode Completion and Step Count: The agent demonstrated the ability to navigate through the game environment, with a gradual increase in the number of steps completed per episode. This indicates an improvement in the agent's understanding and interaction with the game world.

Epsilon (Exploration Rate): The exploration rate started at 1.0 and decayed over time to 0.996, showing a shift from exploration to exploitation as the agent learned from its environment.

Mean Reward and Length: The agent achieved a mean reward of 1436.909 and a mean length of 1483.545 steps over the initial episodes, suggesting a developing proficiency in the game.

The learning progress of the AI agent was monitored through the changes in rewards, loss, and Q-values. Key observations include[11]:

Learning Curves: The agent's learning curves showed a trend of increasing rewards and decreasing losses over episodes, indicating an effective learning process.

Policy Improvement: The agent's policy evolved, showing better decision-making and strategic planning as training progressed. During training, several technical observations were noted:

Environmental Warnings: The training involved handling deprecation warnings and ensuring compatibility with the environment's API.

Video Recording: The ability to record and display gameplay videos was set up, although no specific videos were reported in the results.

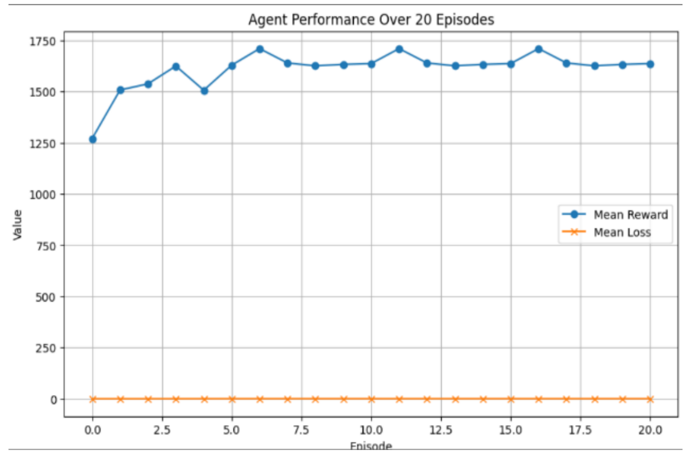

Fig. 3 Agent Performance over Episodes

This line graph visualizes the training performance of an AI agent playing Super Mario Bros over a series of 20 episodes. There are two lines on the graph, each representing different aspects of the agent's performance. The first line, marked with circles ('o'), represents the 'Mean Reward' achieved by the agent in each episode. The second line, marked with 'x', represents the 'Mean Loss' encountered. The x-axis of the graph represents the episode number, ranging from 0 to 20, while the y-axis indicates the values for mean reward and mean loss. The graph includes a legend to distinguish between the two lines, with gridlines for easier reading of values. This graph is crucial for understanding how the agent's ability to maximize rewards and minimize losses evolves throughout the training process.

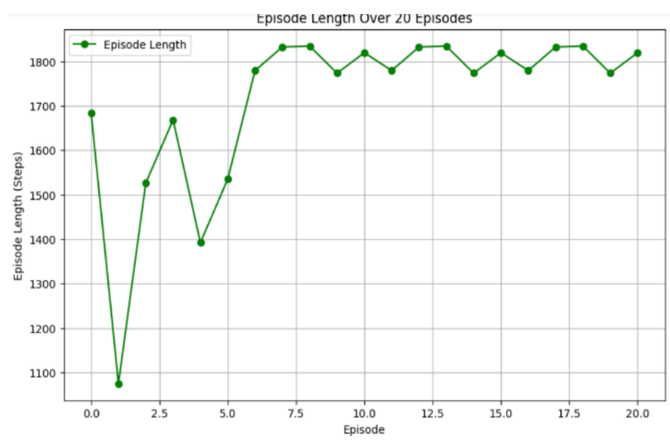

Fig.4 Episode length over Episodes

This line graph shows the length of each episode, measured in steps, over the course of 20 episodes of training the AI agent in Super Mario Bros. The graph has a single line, marked with circles ('o') and colored green, representing the 'Episode Length' for each episode.

Table 1. Agent activity over 20 episodes

Episode	step	Epsilon	Mean Reward	Mean length	Mean loss	Mean-Q Value	Time Delta
0	1683	1.000	1270.000	1683.000	0.000	0.000	22.902
2	3227	0.999	1507.667	1075.667	0.000	0.000	20.897
4	7630	0.998	1537.600	1526.000	0.000	0.000	56.621
6	11682	0.997	1624.571	1668.857	0.143	0.079	58.013
8	12531	0.997	1506.899	1392.333	0.265	0.268	12.781
10	16892	0.996	1628.636	1535.636	0.331	0.412	65.235
12	23132	0.994	1709.231	1779.385	0.370	0.575	94.264
14	27492	0.993	1639.600	1832.800	0.383	0.750	65.496
16	31190	0.992	1625.294	1834.706	0.396	0.911	55.608
18	33690	0.992	1632.105	1773.158	0.412	1.113	37.457
20	38212	0.990	1636.000	1919.619	0.421	1.283	68.739

The x-axis lists the episode numbers from 0 to 20, while the y-axis shows the number of steps taken in each episode. The line graph provides a clear visual representation of how the duration of each episode changes as the agent progresses through its training. An increase in episode length may indicate improved agent performance and ability to navigate the game environment for longer periods, while fluctuations can suggest varying levels of challenge encountered by the agent in different episodes.

In conclusion, the training results underscore the efficacy of the DDQN model in learning and enhancing game performance in the Super Mario Bros environment. The agent's capacity for level navigation, reward optimization, and strategic adaptation over time highlights DRL's potential in complex video game environments

7. CONCLUSION

The research presented in this paper has illuminated the potent capabilities of Deep Reinforcement Learning (DRL) within the realm of artificial intelligence, particularly in the domain of video game playing. By harnessing the complexities and dynamic nature of the iconic Super Mario Bros series, we have demonstrated that DRL, especially through the use of Double Deep Q-Networks (DDQNs), can not only equip AI agents with the skills necessary to navigate and excel in intricate game environments but also potentially exceed human expertise in such tasks.

Our exploration into cutting-edge DRL techniques, including Deep Q-Networks (DQNs), Advantage Actor-Critic (A3C), and Proximal Policy Optimization (PPO), has showcased their effectiveness in cultivating an AI agent capable of strategic decision-making, timing precision, and adaptability—qualities essential for mastering the diverse challenges presented by Mario games. The methodology and experimental setup detailed in this study have underscored the significance of preprocessing techniques and model architecture in enhancing the learning efficiency and performance of DRL models.

The results of our research contribute to the broader understanding of DRL's application in navigating environments that closely mirror real-world complexities and decision-making scenarios. Notably, the agent's improved ability to complete levels, optimize rewards, and strategically adapt over time underscores the potential of DRL in revolutionizing game playing and beyond.

Looking ahead, this study opens several avenues for future exploration, including algorithm optimization, generalization across various games, real-time adaptation, and human-AI collaboration. The findings lay a robust foundation for advancing the field of AI, pushing the boundaries of what is achievable with deep learning technologies.

In conclusion, this research represents a significant step forward in the application of DRL in video gaming, particularly in mastering games as complex as Super Mario Bros. The methodologies, insights, and advancements outlined herein not only bolster the field of AI game playing but also inspire further innovation and exploration in this exciting area of artificial intelligence research.

REFERENCES

1. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533 https://doi.org/10.1038/nature14236
2. Van Hasselt, Hado, Arthur Guez, and David Silver (2016). Deep reinforcement learning with double q-learning. In Proceedings of the AAAI conference on artificial intelligence, 30(1). https://doi.org/10.1609/aaai.v30i1.10295
3. Mnih, Volodymyr, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu (2016). Asynchronous methods for deep reinforcement learning. In International conference on machine learning, 1928-1937. PMLR, https://doi.org/10.48550/arXiv.1602.01783
4. Abdul Aziz, A.(2024). Safe Deep Reinforcement Learning for Super Mario Bros using Heuristics (Doctoral dissertation). https://fse.studenttheses.ub.rug.nl/id/eprint/32359
5. Sumanas, M., Petronis, A., Bucinskas, V., Dzedzickis, A., Virzonis, D. and Morkvenaite-Vilkonciene, I.. (2022). Deep Q-learning in robotics: Improvement of accuracy and repeatability. Sensors, 22(10),.3911. http://dx.doi.org/10.3390/s22103911
6. Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller (2013). Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602. https://doi.org/10.48550/arXiv.1312.5602
7. Zhang, Z., Gao, N., Li, S., Khalid, M.N.A. and Iida, H. (2022). Action Games Evolution Analysis: A Case Study Using the God of War Series. IEEE Access, 10, 123697-123710 .https://doi.org/10.1109/ACCESS.2022.3224469
8. Islam, M.S., Das, S., Gottipati, S.K., Duguay, W., Mars, C., Arabneydi, J., Fagette, A. and Guzdial, M. (2023). Human-AI Collaboration in Real-World Complex Environment with Reinforcement Learning. arXiv preprint arXiv:2312.15160. https://doi.org/10.48550/arXiv.2312.15160
9. Jeerige, A., Bein, D., & Verma, A. (2019). Comparison of Deep Reinforcement Learning Approaches for Intelligent Game Playing. 2019

IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 0366-0371. https://doi.org/10.1109/CCWC.2019.8666545
10. Tampuu, A., Matiisen, T., Kodelja, D., et al. (2017). Multiagent cooperation and competition with deep reinforcement learning. PloS one, 12(4), e0172395. https://doi.org/10.1371/journal.pone.
11. Hausknecht, M., & Stone, P. (2015). Deep recurrent Q-learning for partially observable MDPs. 2015 AAAI Fall Symposium Series. https://doi.org/10.48550/arXiv.1507.06527
12. Xu, Y., Fang, M., Chen, L., Xu, G., Du, Y. and Zhang, C. (2021). Reinforcement learning with multiple relational attention for solving vehicle routing problems. IEEE Transactions on Cybernetics, 52(10), 11107-11120. https://doi.org/10.1109/TCYB.2021.3089179

AUTHORS

M.Nav Deep Undergoing his B.E degree in Computer Science and Engineering from Sathyabama Institute of Science and Technology India from 2020-2024.His areas of interest are Artificial Intelligence, Machine Learning and Image processing.
E-mail: navadeepby99@gmail.com

B.Ankayarkanni received her M.E degree in Computer Science and Engineering from National Engineering College, Kovilpatti,Tamilnadu, India in 2003. and Ph.D degree in Computer Science and Engineering from Sathyabama University India in 2019.Her areas of interest are Artificial Intelligence, Machine Learning and Image processing.
Corresponding author E-mail**:** ankayarkanni.s@gmail.com

J.Albert Mayan received his M.E degree in Computer Science and Engineering from Sathyabama University, Tamilnadu, India in 2005. and Ph.D degree in Computer Science and Engineering from Sathyabama University India in 2019.His areas of interest are Software Testing, Machine Learning and Artificial Intelligence.
E-mail: albertmayan@gmail.com

M.D.Anto Praveena received her M.E degree in Computer Science and Engineering from Sathyabama Institute of Science and Technology Tamilnadu, India in 2008. and Ph.D degree in Computer Science and Engineering from Sathyabama Institute of Science and Technology India in 2022.Her areas of interest are Artificial Intelligence, Machine Learning and Image processing.
E-mail: antopraveena@gmail.com