SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V10I6P104

Volume 10 Issue 6, 27-34, June 2023

Original Article

Self-Driving Electrical Car Simulation using Mutation and DNN

Priyanka Paygude[1], Sonali Idate[2], Milind Gayakwad[3], Namita Shinde[4], Chetan More[5], Amit Patil[6], Rahul Joshi[7], Kalyani Kadam[8], Anand Shinde[9]

[1,2,3,4,5]*Bharati Vidyapeeth (Deemed to be University), College of Engineering, Pune, India.*
[6]*Bharati Vidyapeeth's Jawaharlal Nehru Institute of Technology, Pune*
[7,8]*Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune, India*
[9]*Bharati Vidyapeeth University Institute of Environment Education & Research, Pune*

[1]*Corresponding Author : pspaygude@bvucoep.edu.in*

Received: 26 March 2023 Revised: 19 May 2023 Accepted: 11 June 2023 Published: 30 June 2023

Abstract - *The development of self-driving Electrical Cars has been one of the fascinating fields in the last decade, where machine learning algorithms and neural networks have shown impressive results in enabling autonomous vehicles to perceive and react to their surroundings. However, developing these technologies requires significant hardware, software, and infrastructure investments. This paper presents a self-driving Electrical Car simulation built using neural networks from scratch, without any libraries, using JavaScript. The simulation was developed as a proof of concept to demonstrate that creating a functioning self-driving Electrical Car model is possible using only essential tools and algorithms. The simulation comprises a feedforward neural network that controls the Electrical Car's acceleration and steering force. The Electrical Car is equipped with five sensors that serve as inputs to the neural network, allowing it to navigate through a course without going outside the track.*

Keywords - *Deep Neural Network, Fitness function, Genetic algorithm, Mutation, Self-driving EV.*

1. Introduction

The development of self-driving vehicles has transformed the auto sector and ushered in a new era of mobility. The ability to train and test the automobile's autonomous driving capabilities is crucial to self-driving Electrical Car development. However, using real self-driving Electrical Cars for testing can be risky and expensive. As a result, simulation has become increasingly common in the research and development of self-driving Electrical Car technology. In this study, we offer a simulation of a self-driving Electrical Car created entirely from scratch using neural networks and JavaScript.

Our simulation involves a feedforward neural network trained using a genetic algorithm to learn how to navigate a track without going outside the track boundaries. The neural network takes input from artificial sensors created using rays from the Electrical Car on the canvas. These rays measure the distance to obstacles in a given direction, and the readings of these sensors serve as the input of the Electrical Car's neural network. The neural network output determines the Electrical Car's acceleration and steering force.

We provide the specifics of the self-driving automobile simulation in this study, including the neural network design and the genetic algorithm training procedure. We also examine the prospects for future simulation enhancements and present the simulation's performance on several tracks. The findings of this study show that it is possible to create a self-driving automobile simulation from scratch without using any external libraries and highlight the possibilities for additional study and advancement in this field.

2. Literature Survey

The research is about improvement in the effectiveness and safety of autonomous vehicles in the fast-expanding field of self-driving Electrical Car development [1-3]. This section will discuss recent developments in self-driving Electrical Cars, notably those that employ neural networks and genetic algorithms for control [4]. The research of Pomerleau (1989), who created the ALVINN (Autonomous Land Vehicle in a Neural Network) system, was one of the early efforts on this subject. ALVINN was one of the first successful uses of neural networks for self-driving Electrical Cars [5, 6].

A neural network used in the ALVINN system developed the ability to manoeuvre an Electrical Car using video input [7]. The system was taught using supervised learning by having human experts label samples of the ideal steering angle for a given input image[8, 9].

Recently, scientists have begun investigating the use of genetic algorithms to train neural networks for autonomous vehicles[10-12]. In their study, Chen et al. (2015) trained a feedforward neural network for the steering control of a self-driving Electrical Car using a genetic algorithm [13-15]. A single camera attached to the Electrical Car was the neural network's input, producing the steering angle[16, 17]. The neural network's weights were optimized using a genetic method, with the Electrical Car's fitness function being the distance it travelled before veering off course [18].

Reinforcement learning is a different method of using neural networks to drive self-driving Electrical Cars. An end-to-end system for self-driving Electrical Cars was created by Mnih et al. (2016) using a deep reinforcement learning methodology [19]. The system mastered driving through interactions with the environment and trial-and-error learning [20, 21]. The neural network was trained to forecast the steering angle and the likelihood that the Electrical Car will travel at various speeds [22, 23]. The device could find its way through various situations, including streets and neighbourhoods [24]. [39] A hybrid technique for self-driving automobile control, which combines genetic algorithms with reinforcement learning, was recently proposed in a paper [25, 26]. Convolutional neural networks' topologies were optimized for perception by a genetic algorithm, and control was handled by reinforcement learning [27, 28]. The system showed promise when evaluated on various tasks, including lane following, intersection handling, and roundabout navigation [29, 30].

In this study, we used a genetic algorithm and neural networks to create a simulation of a self-driving Electrical Car [31, 32]. Our simulation was created to show that it is possible to create a working self-driving automobile model entirely from scratch without external libraries [33-35]. In the simulation, the Electrical Car's acceleration and steering forces were controlled by a feedforward neural network, with the network weights developed using a genetic method. Artificial sensors made utilizing rays from the Electrical [36-38] Car on the canvas served as the neural network's input, and the evolutionary algorithm's fitness function was the length [39] of time the Electrical Car stayed on the road before veering off course [40].

The overall potential of neural networks and genetic algorithms in the creation of self-driving Electrical Cars is demonstrated by the works evaluated in this literature overview [41-43]. These methods have the potential to completely transform the automobile industry in the following years since they have shown encouraging results in enabling autonomous vehicles to detect and respond to their surroundings [44].

3. Methodology

The methodology used in this research involves the development of a self-driving Electrical Car simulation built using Reinforcement learning and trained with a genetic algorithm. The simulation is implemented in JavaScript without the use of external libraries. This section explains the algorithm, including the parameters, formulas, and procedures involved.

3.1. Problem Formulation

The self-driving Electrical Car simulation aims to teach a neural network how to travel along a track without straying outside its borders. The acceleration and steering force of the Electrical Car is managed by its steering force, which is dictated by the neural network's output. Artificial sensors that measure the separations between the Electrical Car and obstructions in its path provide the neural network's inputs.

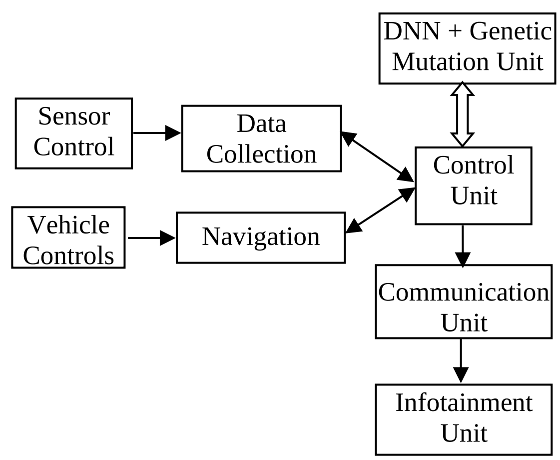

Fig. 1 Core architecture of self-driving EV

3.2. Neural Network Architecture

During the simulation, a fully connected, feedforward neural network was deployed. This structure comprises a hidden layer, an output layer, and an input layer. Seven neurons make up the input layer, which corresponds to the input features of the automobile and includes the false sensor signals. The acceleration and steering forces are represented by the hidden layer's ten neurons and the output layer's two neurons.

As mentioned in Fig.1, the overall controlling and computation are performed using the Genetic model and Deep Neural Network. The term mutation is used as the environment across the car tends to change according to time, situation, and geographical location. The models must be self-mutating to cope with the changes. The significance of the DNN is to deal with the data collected through various sensors like Radar, Lidar, and cameras and process the information to make decision-making easy. The output can be seen as actual vehicle Control accession through the communication unit. The data visualisation is possible at the front desk using an infotainment unit. Human assessment is also needed if the decision is altered while driving the self-driving electric car. The research article covers the simulation of the model mentioned above.

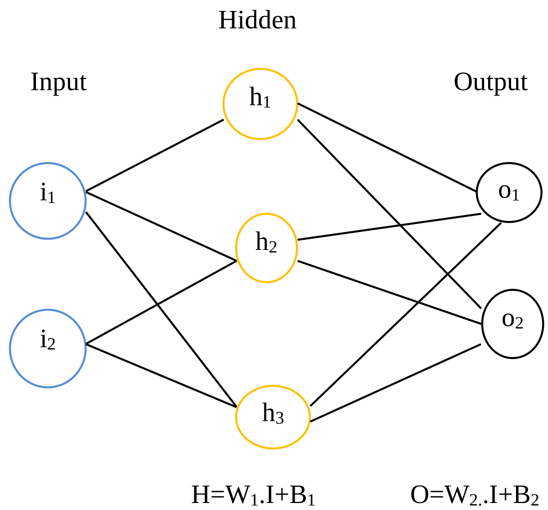

Fig. 2 Basic neural architecture

3.3. Genetic Algorithm

The neural network weights are trained using a genetic method. To improve the self-driving Electrical Car's performance, it simulates the process of natural selection. The genetic algorithm entails the following crucial steps:

- Initialization: A population of N randomly initialized Electrical Cars is spawned. Each Electrical Car has its set of neural network weights.
- Evaluation: Each Electrical Car in the population is evaluated by running it through the track using its current set of weights. The fitness of each Electrical Car is determined based on its performance, such as the distance travelled without going off the track.
- Selection: Two random parents are selected from the generation. Their genome is used to produce an Electrical Car using random values from each parent.
- Crossover: The selected Electrical Cars are recombined with each other to create new "offspring" Electrical Cars. Crossover is performed by exchanging genetic material (i.e., weights) between the parents to produce diverse offspring.
- Mutation: The offspring of Electrical Cars undergo slight random mutations to introduce further diversity into the population. This helps prevent premature convergence and allows the exploration of different weight configurations.
- Repeat: The newly created population of Electrical Cars goes through the evaluation, selection, crossover, and mutation steps iteratively. This process constitutes one generation.
- Stopping Condition: The genetic algorithm continues to run for a predetermined number of generations or until a satisfactory level of performance is achieved.

Pseudocode for the Genetic Algorithm:

```
procedure GeneticAlgorithm():

    // Initialize a population of N Electrical Cars
    population = InitializePopulation(N)

    repeat until termination condition is met:

    // Evaluate the fitness of each Electrical Car
    fitnessScores = EvaluatePopulation(population)

    // Select parents for reproduction
    parents = Selection(population, fitnessScores)

    // Perform crossover to create offspring
    offspring = Crossover(parents)

    // Apply mutation to introduce diversity
    mutatedoffspring = Mutation(offspring)

    // Replace the old population with the new one
    population = ReplacePopulation(population,
        mutatedOffspring)

        return population
```

3.4. Training Process

The technique used to train the Electrical Car is called Reinforcement learning. We used two criteria to judge the Electrical Car's performance and created reward and penalty rules that the Electrical Car had to follow:

To do:
1. Reach the finish line (+x points)
2. Drive fast (+y points)

Not to do:
1. Go outside the track (-z points)
2. Drive very slow (-w points)

For this research, we allotted points to each Electrical car depending on the number of checkpoints it crossed on the track multiplied by 100. For the successful track completion, we rewarded each Electrical Car with 500 points, and if the

Electrical Car went outside the track, it was given a penalty of one minute.

Using this, the fitness of each Electrical Car is calculated by the formula:

Fitness = Points / Time Factor

Because of this, the Electrical Car that will reach the finish line will have higher points than those that did not. Also, Electrical Cars which will reach the finish line faster would have greater points than the slow ones.

The next generation is created using random genes from the first generation using a Genetic algorithm. The sum of the fitness of each Electrical Car is calculated from the first generation; after that, a random number is picked from 0 to S (S being the sum of finesses). Then we iterate over citizens adding the fitness of each, when the sum is greater than the random number, we select the parent. Both parents are selected with this method. The uniform crossover technique produces an Electrical Car with both genes from its parents. The mutation is introduced for the newly produced Electrical Car to discover new solutions. The offspring produced by the parents are not guaranteed to have better genes. To solve this problem, we used Elitism to save the best-performing Electrical Car, i.e., Electrical Car with the highest points for the next generation.

During each generation, the neural network weights are updated based on the fitness of the Electrical Cars. The weights are updated using the formula:

$$\Delta w = \alpha * f(x) * r * (\mu - f(x))$$

$$w' = w + \Delta w$$

Where Δw is the weight change, α is the learning rate, $f(x)$ is the fitness of the Electrical Car, r is a random number between 0 and 1, and μ is the population's mean fitness. The learning rate determines the step size in the weight update process, and it controls the balance between exploration and exploitation. A faster convergence may result from greater weight updates made possible by a higher learning rate, but the chance of overshooting optimal solutions also rises. The formula ensures that updates to weights with higher fitness values are smaller than changes with lower fitness values. This approach permits investigating different solutions while promoting the transmission of advantageous weight configurations.

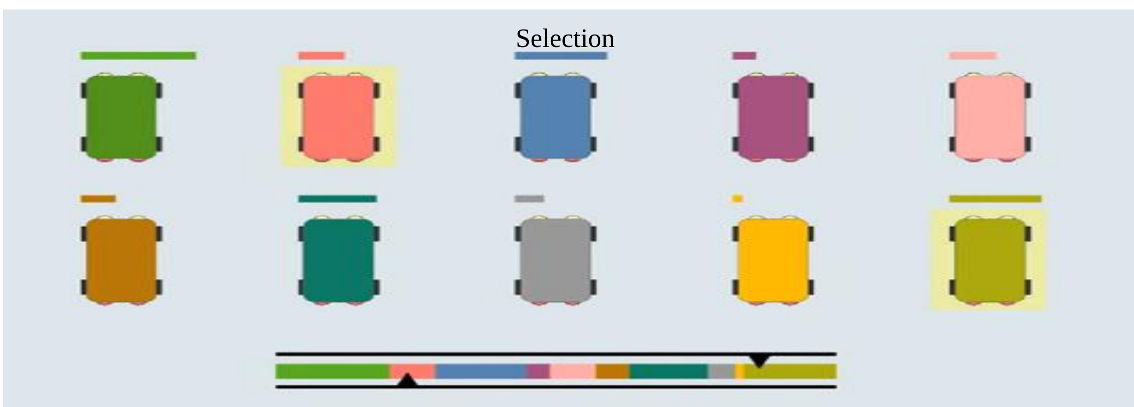

Fig. 3 Selection of parents

Fig. 4 Uniform crossover with mutation

3.5. Parameter Tuning

Several parameters must be tweaked to get the best configuration for the self-driving automobile simulation. The size of the population, the number of generations, the mutation rate, the likelihood of a crossover, the neural network architecture, and the learning rate are some of these parameters. The values used for these parameters substantially influence the self-driving Electrical Car simulation's performance.

A systematic approach is followed to determine the best values for the parameters. A range of values is defined for each parameter, and multiple experiments are conducted with different combinations of parameter values. The performance of the simulation is evaluated based on metrics such as the distance travelled without going off the track or the time taken to complete the track.

The following chart summarizes the different parameters used in the algorithm, their values, and the best result achieved with each value:

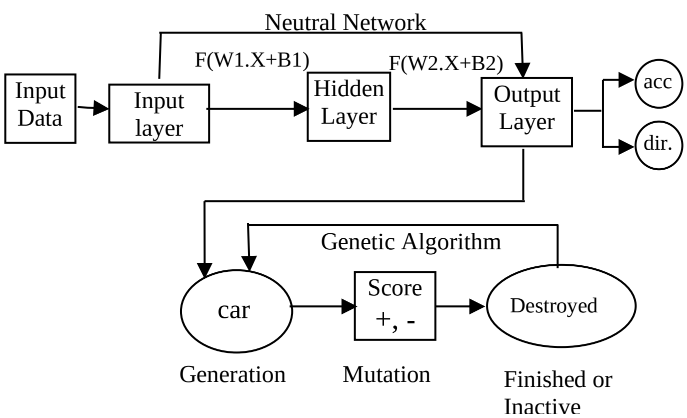

Fig. 5 Architectural diagram of the training

Table 1. Parameters used in training

Parameter	Values	Best Result Achieved
Population Size	50, 100, 200	200
Number of Generations	50, 100, 200	200
Mutation Rate	0.01, 0.05, 0.1	0.05
Crossover Probability	0.6, 0.8, 1.0	0.8
Learning Rate	0.001, 0.01, 0.1	0.01
Neural Network Architecture	[7-10-2], [9-12-2], [10-15-2]	[10-15-2]

These values were obtained through rigorous experimentation and analysis. A larger population size and a higher number of generations allowed more exploration and improved convergence to optimal solutions. A moderate mutation rate ensured sufficient diversity in the population, preventing premature convergence. The crossover probability of 0.8 promoted the exchange of genetic material, facilitating the discovery of better weight configurations-a learning rate of 0.01 balanced exploration and exploitation, leading to steady progress during training. The [10-15-2] configuration consistently produced the best results among the tested neural network architectures[46-50].

3.6. Training and Evaluation

The self-driving Electrical Car simulation is trained using the defined parameters and the genetic algorithm. Each Electrical Car in the population undergoes multiple evaluations, selection, crossover, and mutation generations. The fitness of the Electrical Cars improves over time as the neural network weights are adjusted through the genetic algorithm.

The simulation continues until a stopping condition is met, such as reaching a predefined fitness threshold or completing the track successfully. After training, the performance of the self-driving Electrical Car is evaluated on various tracks. The evaluation includes metrics such as the completion time, the number of track boundary violations, and the smoothness of the Electrical Car's movements. The trained self-driving Electrical Car is tested on known and unseen tracks to assess its generalization and adaptability.

3.7. Performance Analysis

The performance of the self-driving Electrical Car simulation is assessed based on the achieved results. Metrics such as the average distance travelled without going off the track, the success rate in completing the track, and the efficiency of the Electrical Car's movements are analyzed. Additionally, comparisons may be made with other existing self-driving Electrical Car models or algorithms to gauge the effectiveness and competitiveness of the developed simulation.

The strengths and limitations of the self-driving Electrical Car simulation can be identified by conducting thorough experiments and analysing the performance metrics. The findings contribute to understanding the effectiveness of the chosen methodology and provide insights for further improvements and future research.

In conclusion, the self-driving Electrical Car simulation methodology uses a genetic algorithm to train a neural network. The algorithm incorporates parameter tuning, systematic training, and performance analysis. The developed chart summarizes the different parameters, their values, and the best results achieved with each value. This approach optimises the self-driving Electrical Car simulation and provides a foundation for further research and development in autonomous vehicles.

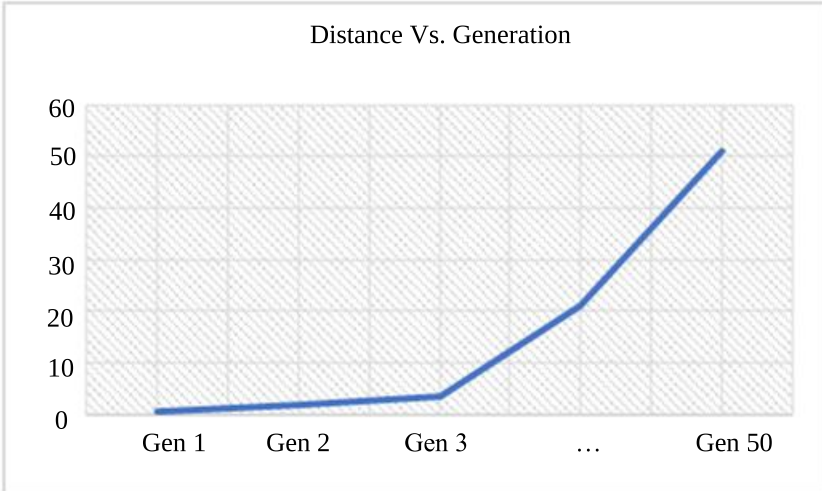

Fig. 6 Graph showing distance max travelled in each generation

4. Conclusion

The self-driving Electrical Car simulation presented in this paper achieved impressive results regarding the Electrical Car's ability to navigate through various tracks without going outside the track boundaries. The simulation was run for 50 generations, and the best-performing Electrical Car completed all tracks. The neural network architecture and the genetic algorithm effectively trained the Electrical Car to navigate the courses.

One of the main advantages of using simulation for self-driving Electrical Car development is the reduced risk and cost associated with physical testing. The simulation presented in this paper provides a cost-effective and safe alternative to physical testing, allowing developers to test and improve their algorithms in a virtual environment. Using JavaScript without external libraries demonstrates that creating a functioning self-driving Electrical Car model is possible using only essential tools and algorithms.

Acknowledgement

Vaibhav Shukla, Ashutosh Kumar Singh, and Arvind Kumar Jha helped simulate a self-driven electrical car.

References

[1] Huang H, and Sun S, "A Survey on Self-driving Electrical Cars: Advances, Challenges, and Future Directions," *IEEE Access*, vol. 9, pp. 121324-121346, 2021.

[2] Wen-Kung Tseng, and Hou-Yu Chen, "The Study of Tracking Control for Autonomous Vehicle," *SSRG International Journal of Mechanical Engineering*, vol. 7, no. 11, pp. 57-62, 2020. [CrossRef] [Publisher Link]

[3] Priyanka G. Gonnade, "Vehicle Tracking and Transportation Management System," *International Journal of Computer and Organization Trends*, vol. 10, no. 1, pp. 1-4, 2020. [CrossRef] [Publisher Link]

[4] Mariusz Bojarski et al., "End to End Learning for Self-Driving Electrical Cars," *arXiv preprint arXiv:1604.07316*, 2016.

[5] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep Learning," *Nature*, vol. 521, pp. 436-444, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[6] Timothy P. Lillicrap et al., "Continuous Control with Deep Reinforcement Learning," *arXiv preprint arXiv:1509.02971*, pp. 1-14, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[7] Volodymyr Mnih et al., "Human-Level Control through Deep Reinforcement Learning," *Nature*, vol. 518, pp. 529-533, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[8] Dean A. Pomerleau, "Efficient Training of Artificial Neural Networks for Autonomous Navigation," *Neural Computation*, vol. 3, no. 1, pp. 88-97, 1991. [CrossRef] [Google Scholar] [Publisher Link]

[9] Jianjun Nie et al., "A Survey on Theories and Applications for Self-Driving Cars Based on Deep Learning Methods," *Applied Science*, vol. 10, no. 8, pp. 1-29, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[10] Siyuan Chen et al., "Stabilization Approaches for Reinforcement Learning-Based End-to-End Autonomous Driving," *In IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 4740-4750, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[11] Xiang Zhang, Junbo Zhao, and Yann LeCun, "Character-Level Convolutional Networks for Text Classification," *Advances in Neural Information Processing Systems 28 (NIPS)*, pp. 649-657, 2015. [Google Scholar] [Publisher Link]

[12] Abhishek Varshney et al., "Study of Solving Knapsack Problem using Genetic Algorithm," *Solid State Technology*, vol. 63, no. 2s, pp. 93-99, 2020. [Google Scholar] [Publisher Link]

[13] Christian Birchler et al., "Machine Learning-Based Test Selection for Simulation-Based Testing of Self-Driving Cars Software," *Empirical Software Engineering*, vol. 28, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[14] Fei Ye et al., "A Survey of Deep Reinforcement Learning Algorithms for Motion Planning and Control of Autonomous Vehicles," *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1073-1080, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[15] P. Mahalakshmi, and R. Vallikannu, "New Miniature Dual Mode Rat-Race Coupler Design for Autonomous Vehicles," *International Journal of Engineering Trends and Technology*, vol. 70, no. 3, pp. 222-233, 2022. [CrossRef] [Publisher Link]

[16] Chenyi Chen et al., "DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2722-2730, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[17] Ian Goodfellow et al., "Generative Adversarial Nets," *In Advances in Neural Information Processing Systems 27 (NIPS)*, pp. 2672-2680, 2014. [Publisher Link]

[18] Kaiming He et al., "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[19] Brody Huval et al., "An Empirical Evaluation of Deep Learning on Highway Driving," *arXiv preprint arXiv:1504.01716*, pp. 1-7, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[20] Li Y et al., "Deep Reinforcement Learning-Based Decision-Making System for Autonomous Driving," *IEEE Access*, vol. 5, pp. 21717-21727, 2015.

[21] Xinlei Pan et al., "Virtual to Real Reinforcement Learning for Autonomous Driving," *arXiv preprint arXiv:1704.03952v4*, pp. 1-13, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[22] Arman Asgharpoor Golroudbari, and Mohammad Hossein Sabour, "Recent Advancements in Deep Learning Applications and Methods for Autonomous Navigation: A Comprehensive Review," *arXiv preprint arXiv: arXiv:2302.11089v3*, pp. 1-56, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[23] David Silver et al., "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, vol. 529, pp. 484-489, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[24] Zhang Z et al., "Speed/Steering Control of Self-Driving Electrical Cars in the Presence of Limited Information," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 11, pp. 9809-9822, 2017.

[25] Kavita K. Beldar, M. D. Gayakwad, and M. K. Beldar, "Optimizing Analytical Queries on Probabilistic Databases with Unmerged Duplicates using MapReduce," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 4, no. 5, pp. 9651-9659, 2016. [Google Scholar] [Publisher Link]

[26] Kavita K. Beldar et al., "Query Evaluation on Probabilistic Databases using Indexing and MapReduce," *International Journal of Database Theory and Application*, vol. 9, no. 10, pp. 363-378, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[27] Kavita K. Beldar et al., "A Comparative Analysis on Contingence Structured Data Methodologies," *International Journal of Software Engineering and Its Applications*, vol. 10, no. 5, pp. 13-22, 2016. [Google Scholar] [Publisher Link]

[28] Menka K. Beldar et al., "Altruistic Content Voting System using Crowdsourcing," *International Journal of Scientific Research and Review*, vol. 7, no. 5, pp. 477-486, 2018. [Google Scholar] [Publisher Link]

[29] Menka K. Beldar et al., "Survey on Classification of Online Reviews Based on Social Networking," *International Journal on Future Revolution in Computer Science & Communication Engineering*, vol. 4, no. 3, pp. 250-253, 2018. [Publisher Link]

[30] Milind Digambar Gayakwad, "Survey on Inappropriate Content Detection in Online Social Media," *International Journal of Innovative Research in Science, Engineering and Technology*, vol. 8, no. 9, pp. 9297-9302, 2022. [Publisher Link]

[31] Gayakwad M. D, and Phulpagar B. D, "Research Article Review on Various Searching Methodologies and Comparative Analysis for Re-Ranking the Searched Results," *International Journal of Recent Scientific Research*, vol. 4, no. 11, 1817-1820, 2013. [Google Scholar] [Publisher Link]

[32] Pankaj D. Khambre, Amit Kumar, and M. D. Gayakwad, "VLAN Implementation using IP over ATM," *Journal of Engineering Research and Studies*, vol. 2, no. 4, pp. 186-192, 2011. [Google Scholar] [Publisher Link]

[33] Milind Gayakwad, and Suhas Patil, "Content Modelling for Unbiased Information Analysis," *Library Philosophy and Practice*, pp. 1-17, 2020. [Google Scholar] [Publisher Link]

[34] Milind Gayakwad, and Suhas Patil, "Analysis of Methodologies to Model the Content for Conveying the Correct Information," *2021 International Conference on Computing, Communication and Green Engineering (CCGE)*, pp. 1-4, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[35] Paule Kevin Nembou Kouonchie, Vitalice Kalecha Oduol, and George Nyauma Nyakoe, "Roadside Unit Transmission Control for Energy Efficiency in Vehicle-to-Infrastructure Communication Network," *International Journal of Engineering Trends and Technology*, vol. 70, no. 5, pp. 145-158, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[36] Milind Gayakwad, and Suhas Patil, "Assessment of Source, Medium, and Intercommunication for Assessing the Credibility of Content," *2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*, pp. 1-5, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[37] Milind Gayakwad et al., "Credibility Evaluation of User-Generated Content using Novel Multinomial Classification Technique," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 10, no. 2s, pp. 151-157, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[38] Milind Gayakwad et al., "Credibility Analysis of User-Designed Content using Machine Learning Techniques," *Applied System Innovation*, vol. 5, no. 2, pp. 1-15, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[39] Gajanan Bhole, Swati T. Harane, and Milind Gayakwad, "Secure Search over Encrypted Data Techniques: Survey," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 7, pp. 575-577, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[40] Kavita Santosh Shevale, Gajanan V. Bhole, and Milind Gayakwad, "Literature Review on Probabilistic Threshold Query on Uncertain Data," *International Journal of Current Research and Review*, vol. 9, no. 6, pp. 52482-52484, 2017. [Publisher Link]

[41] Mahamat Adam Boukhari, and Milind Gayakwad, "An Experimental Technique on Fake News Detection in Online Social Media," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 8, no. 8S3, pp. 526-530, 2019. [Google Scholar] [Publisher Link]

[42] Maruti Maurya, and Milind Gayakwad, "People, Technologies, and Organizations Interactions in a Social Commerce Era," *In Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI-2018)*, pp. 836-849, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[43] Milind Gayakwad, and B. D. Phulpagar, "Requirement Specific Search," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 11, p. 121, 2013.

[44] Iftikhar Ahmad et al., "Fake News Detection using Machine Learning Ensemble Methods," *Complexity*, vol. 2020, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[45] Priyanka Paygude, and Aatmic Tiwari, "Comparative Analysis on Different Generations of Cryptocurrency," *International Journal of Future Generation Communication and Networking*, vol. 14, no. 1, pp. 3016-3026, 2021. [Publisher Link]

[46] Priyanka Paygude, and P. R. Devale, "Automated Data Validation Testing Tool for Data Migration Quality Assurance," *International Journal of Modern Engineering Research (IJMER)*, vol. 3, no. 1, pp. 599-603, 2013. [Google Scholar] [Publisher Link]

[47] Priyanka Paygude, Shashank D. Joshi, and Manjusha Joshi, "Fault Aware Test Case Prioritization in Regression Testing using Genetic Algorithm," *International Journal of Emerging Trends in Engineering Research*, vol. 8, no. 5, pp. 2112-2117, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[48] Priyanka Paygude, and Shashank D. Joshi, "Use of Evolutionary Algorithm in Regression Test Case Prioritization: A Review," *Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2018)*, vol. 31, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[49] Priyanka Paygude, and Shashank Joshi, "Optimization of Test Case Prioritization through Calibration of Genetic Algorithm," *Solid State Technology*, vol. 63, no. 1s, pp. 2377-2386, 2020. [Google Scholar] [Publisher Link]

[50] Priyanka Paygude et al., "Comparative Analysis of Stock Prices by Regression Analysis and FB Prophet Models," *Data Intelligence and Cognitive Informatics*, pp. 295-307, 2022. [CrossRef] [Google Scholar] [Publisher Link]