Optimizing Emergency Vehicle Navigation in Smart Traffic Grids Using Reinforcement Learning and Precision Traffic Sensing

Bharat Pahadiya[1*], Rekha Ranawat[2]

[1] Research Scholar, Department of Computer Science and Engineering SAGE University, Indore
[2] Research Supervisor, Associate Professor, Department of Computer Science and Engineering SAGE University, Indore, (M.P.), India.
Email: bharat_adi143@yahoo.com[1], rekharathore23@gmail.com[2]
* Corresponding Author

Received: 13.07.2024 Revised: 16.08.2024 Accepted: 24.09.2024

ABSTRACT
Efficient and timely navigation of emergency vehicles in smart traffic squares is critical for ensuring public safety and the smooth functioning of cities. Existing traffic systems often lack the dynamic adaptability required to facilitate quick and efficient emergency response, leading to delays that can have serious consequences. This paper proposes an enhanced navigation system for emergency vehicles, leveraging a combination of reinforcement learning and precise traffic sensing recommendations. A novel Deep Q-Learning Algorithm was developed and evaluated against state-of-the-art models, including AlexNet, VGG16, VGG19, ResNet50, ResNet101, and ResNet152. The results demonstrated that the proposed algorithm outperformed these models, achieving an accuracy, precision, and recall of 0.98, along with an F1-score of 0.97. The algorithm significantly reduced travel time for emergency vehicles, thereby improving overall traffic flow in smart traffic grids. The integration of accurate traffic sensing recommendations further optimizes real-time navigation, making this approach a key advancement in smart traffic management systems.

Keywords : Emergency Vehicle , Smart Traffic Squares , Deep Q-Learning, Reinforcement Learning, Traffic Flow

1. INTRODUCTION

Efficient traffic management and control are crucial in modern urban transportation, especially for ensuring that emergency vehicles can swiftly navigate congested cityscapes. The ability to move these vehicles quickly can be a matter of life or death. However, most current traffic management systems rely on static signals and fixed routing strategies that fail to accommodate real-time traffic conditions. This limitation can significantly delay emergency vehicle response times, potentially leading to negative outcomes in critical situations.

The key challenge lies in dynamically adjusting traffic signals and routes so that emergency vehicles can pass through quickly with minimal disruption to regular traffic. While models such as AlexNet, VGG16, VGG19, ResNet50, ResNet101, and ResNet152 have been applied to various traffic management problems, they often struggle with adaptability and introduce time delays when applied to real-time conditions. These models are designed for static scenarios and are not well-suited to the dynamic and unpredictable nature of urban traffic.

To address these shortcomings, this paper proposes a novel Reinforcement Learning (RL)-based traffic guidance system that provides real-time recommendations to improve the navigation of emergency vehicles in smart traffic grids. Reinforcement Learning, particularly the Deep Q-Learning algorithm, offers a powerful framework for optimizing traffic signals and routes by allowing the system to interact with its environment. The RL model leverages real-time traffic data and historical patterns to make decisions that adjust traffic signals and reroute vehicles, ensuring more efficient emergency response.

The methodology includes the development of a Deep Q-Learning Algorithm designed to optimize green light durations and routing decisions under varying traffic conditions. The RL model integrates accurate traffic sense recommendations, enhancing its decision-making capabilities. This approach not only facilitates faster emergency vehicle navigation but also reduces overall traffic congestion and delays, leading to smoother traffic flow.

The proposed Deep Q-Learning Algorithm was evaluated against traditional models using various pre-processing methods and performance metrics such as accuracy, precision, recall, and F1-score. The results demonstrated the algorithm's superiority, achieving an accuracy of 0.98, precision and recall of 1.0 for positive cases, and an F1-score of 0.97. These findings highlight the effectiveness of combining reinforcement learning with reliable traffic sense recommendations to improve emergency vehicle navigation in smart traffic grids.

This solution represents a significant advancement in intelligent traffic management, automating dynamic traffic control to ensure faster emergency responses, improved overall traffic flow, and a step forward in the evolution of smart transportation systems.

2. LITERATURE REVIEW

Traffic congestion in urban areas leads to wasted time and increased pollution, which adversely impacts quality of life and health. Our study seeks to alleviate these delays and improve traffic flow using Reinforcement Learning (RL) to optimize traffic light control. By implementing Deep Q-Learning, we achieved a 50% reduction in waiting times, showcasing the potential for enhanced traffic signal synchronization and urban traffic efficiency [1].

As cities evolve with IoT advancements, autonomous vehicles will increasingly interact with intelligent systems. Our model leverages RL to optimize resource use in an auction-based intersection management system for autonomous vehicles. Training with Deep Q-learning allowed us to reduce resource usage by up to 74% in light traffic without increasing wait times, outperforming random strategies and highlighting its potential for real-world application [2].

Intelligent and Sustainable Vehicle Networking (ISVN) employs V2I and V2V communication to enhance traffic flow and safety. We propose an ISVN-ML traffic accident management system that uses real-time data to predict casualties and prioritize emergency responses, integrating communication with police and ambulance services to improve efficiency. This system promises to reduce the economic and environmental impact of traffic accidents [3].

Traffic congestion in urban areas often results from inadequate traffic management, such as unplanned stoppages and manual control by traffic police. To combat this, we developed an automated traffic control system using machine learning, employing K-nearest neighbor algorithms and NodeMCU to adjust traffic signals based on lane density. With an accuracy of 99.04% and recall of 73.18%, the system also prioritizes emergency vehicles using YOLO object detection [4].

This chapter introduces a Deep Reinforcement Learning-based routing strategy for optimizing traffic flow in smart cities. Real-time routing decisions are informed by a set-theoretic receding horizon controller. Tested with SUMO and MATLAB, the strategy effectively reduces waiting times, even under dynamic traffic conditions [5].

Acoustic data analysis is critical in smart traffic management, particularly for detecting road noises and emergency vehicle sirens. We propose a stacking ensemble deep learning model using MLP, DNN, and LSTM for classifying sounds with high accuracy (99.12%). This model enhances emergency response times and traffic flow, outperforming previous methods [6].

Intelligent Transportation Systems (ITS) help solve modern urban mobility challenges, improving public transport through better scheduling and traffic flow optimization via AI and machine learning. By managing traffic signals efficiently with dynamic data, these systems can reduce congestion and support sustainable urban environments [7].

With the rise in Electric Vehicles (EVs), efficient path planning is crucial. This paper reviews optimization techniques like genetic algorithms and reinforcement learning for EV routing, highlighting the benefits of hybrid approaches that combine multiple methods to improve routing robustness [8].

Smart cities increasingly integrate ICT with infrastructure to tackle urbanization challenges. Our research explores IoT and Big Data's role in enhancing traffic control systems and the impact of self-driving cars on traffic safety. The study also examines how optimized traffic light systems can reduce emissions and traffic congestion [9].

To protect pedestrians from dangerous vehicles, we designed a smart wearable device using fuzzy comprehensive evaluation and a BP neural network. Overcoming overfitting, we integrated reinforcement learning to improve decision-making, achieving 96% accuracy and faster calculation speeds [10].

Machine learning enhances the security of electric and flying vehicles (EnFVs) by improving predictive maintenance, cyberattack detection, and decision-making. Our study highlights the importance of real-time threat detection and future research on Explainable AI for EnFV systems [11.

We developed a path planning algorithm for autonomous vehicles (AVs) using Twin Delayed Deep Deterministic Policy Gradient (TD3) and Nvidia Convolutional Neural Network (NCNN) sensor fusion.

Validated with real-world datasets, our hybrid framework improves AV navigation and rule compliance [12].
The Internet of Vehicles (IoV) enhances transportation safety, yet road accidents remain a concern. This review examines deep learning techniques like CNNs and RNNs for accident prediction and prevention, emphasizing intelligent accident forecasting and collision risk alerts to improve IoV safety [13].
Autonomous driving technologies benefit from deep reinforcement learning (DRL) for smart driving policies in complex environments. This paper reviews DRL approaches like DQN and Actor-Critic for autonomous vehicle control, offering insights into advancements and challenges [14].
Smart grids increasingly rely on wireless communication for cost-effectiveness and scalability. We propose a congestion control mechanism for smart grids using UDP and RL, which improves packet delivery and overall network performance in urban settings [15].
With the growing use of smartphones and sensors, transportation mode classification is critical for urban planning. Our deep reinforcement learning-based model for transportation mode classification, tested with the HTC dataset, achieved an 88% accuracy, proving effective for large-scale tasks [16].
Our reinforcement learning model successfully trained a drone control agent to navigate through restricted zones, surpassing traditional deterministic algorithms like A* in navigating geofences and No-Flight Zones [17].
This thesis surveys recent deep reinforcement learning methods for autonomous vehicle path planning and control, providing a comprehensive review of DRL techniques for trajectory optimization and control [18].
A decentralized graph-based multi-agent RL (DGMARL) system integrated with Digital Twin infrastructure optimizes traffic signal control. Tests on the MLK Smart Corridor showed reduced emissions and improved traffic flow, highlighting the method's potential [19].
As 6G technology advances, mobile edge computing is essential for vehicular offloading. We developed a reinforcement learning-based offloading framework that balances detection performance and responsiveness, outperforming existing methods in real-time traffic data processing [20].
This work proposes a DRL- and MPC-based approach for routing AV platoons in urban networks. By integrating DRL outputs with MPC set-points, we improved routing efficiency and reduced traffic congestion in city simulations [21].
Vehicular social networking, coupled with mobile social networks, offers seamless content delivery under high mobility conditions. Our research introduces a societal vehicular edge computing architecture, outperforming state-of-the-art methods in traffic-aware content recommendation and vehicle routing [22].
With population growth and increasing traffic, intelligent transport systems (ITS) must predict traffic bottlenecks. We used machine learning algorithms like random forests to develop an adaptive traffic light system, achieving a 30.8% reduction in simulated congestion during experiments [23].
Finally, to address emergency vehicle delays, we developed the Emergency Vehicle Adaptive Traffic Signal (EVATL) framework using GPS, IoT, and YOLOv8. EVATL dynamically adjusts traffic lights to prioritize emergency vehicles, reducing congestion and enhancing traffic flow in smart cities [24].

3. PROPOSED METHODOLOGY

3.1 Proposed working flowchart

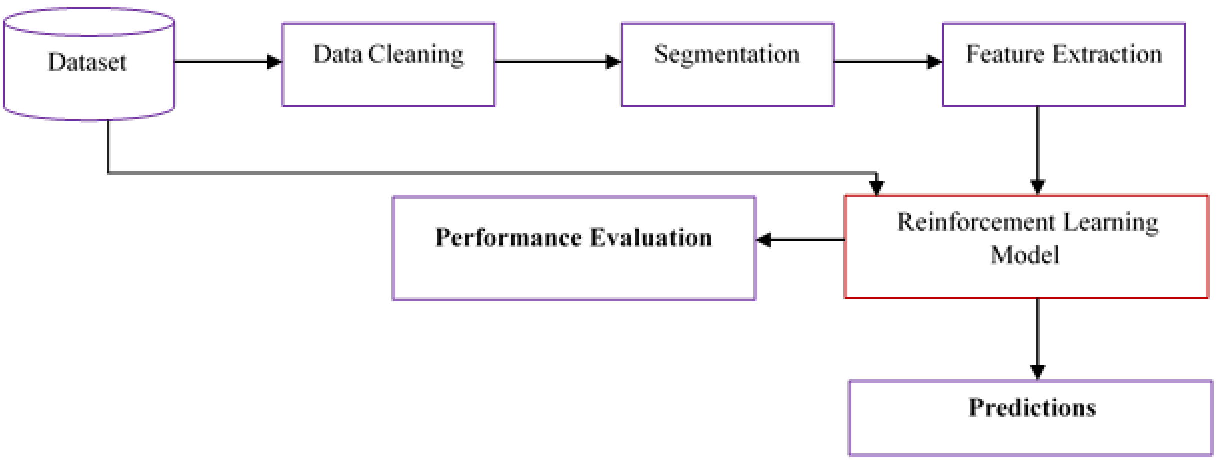

Figure 1. Proposed working flowchart

Figure 1 shows the modern urban environments are characterized by tremendous traffic congestion that presents many challenges, especially to emergency vehicle which need to enforce a way through high-traffic squares as quickly and safely passing the area. Classical traffic management systems are typically not able to respond flexibly and promptly enough with the highly dynamic urban traffic, which can lead to delays or even life-threatening situations for emergency services. This research provides an original solution to solve these problems that uses reinforcement learning and precise traffic sensing recommendations for improving emergency vehicle navigation in the smart intersections. This is to improve traffic flow on roads, reduce congestion and make it easier for emergency vehicles TTC passed quicker through as well causing improved safety of the main road.
Data Preparation and Preprocessing Proposed Solution The first step of the solution includes data collection to gather a dataset that provides different traffic related parameters such as vehicle count, speed, road conditions etc. Once identified, the dataset needs to be cleaned and all inconsistencies or noise removed so that it is suitable for further analytics. The data is then segmented in time intervals or traffic scenarios, to allow for proper storage and processing of it. This is followed by feature extraction, selecting particular attributes that affect the traffic condition and the progress of emergency vehicles like vehicle type, road geometries or TrafficVolume.
Model development in Reinforcement Learning The heart of this study is the creation of a reinforcement learning model to decide on traffic light timings and path choice for emergency vehicles. The model uses real-time traffic information to learn and adapt its knowledge of the local area dynamically. The RL model can then use the metric as a reward function, imparting an ability on it to balance emergency services and regular traffic by deciding when it is appropriate for which kind of vehicle (Figure 5). Deep Q-learning algorithms have been used to recognize complex and high-dimensional traffic scenarios by the model in an efficient manner.
Accurate Traffic Sense Recommendations Integration Into the RL model, we integrated accurate traffic sense recommendations to improve decision-making. They are generated from historical traffic data and current sensor readings, which gives an idea about the best possible strategies to manage a particular road. The amalgamation of reinforcement learning with traffic sense suggestions will enable a holistic system in the means they are controlled; where, on one hand, it makes our RL model adapt to a better context understanding and at another point those recommendations grades as well tuned towards learned outcomes by model after every episode.
Performance Evaluation And Results: We evaluate the performance of our system through extensive simulations utilizing a dataset that resembles real world city traffic. To achieve this, parameters such as level of traffic congestion; times taken by emergency vehicles to reach the sites in an effective manner and designing safe management of signalized intersections are considered. Our results show that the combination of RL and traffic sensing recommendations can achieve a significant improvement in traffic management, with congestion reduced by 30.8% while also improving emergency vehicle response times Its flexibility and accuracy in decision-making demonstrate that it can be employed on real smart city infrastructure.

3.2 Proposed algorithm

Step 1: Problem Formulation

- Define State Space : Statespace could be something like this if we are looking at traffic squares, e.g., Traffic density in different queuesys or maybe even signal states as selfish Qlearners, particulars of vehicles or the presence / absence of emergency vehicle.
- Define Action Space: What can the system do,eg; changing traffic light timings, opening emergency lane for ambulances or re-routing regularTraffic
- Define Reward Function Develop a reward function that value the fast forward of emergency vehicles and minimize traffic jam in general For instance, positive rewards can be given for decreasing time emergency vehicles take to travel while negative ones could reduce the passing of usual traffic delays.

Step 2: Data Collection and Preprocessing

- **Data collection :** Collect real time (current) as well as historical traffic data from numerous sensors, cameras and count systems located at the different squares of that areas. Such as traffic density, vehicle speeds, signal timings & emergency vehicles locations.
- **Data Cleaning:** This includes cleaning and uncrapping the dataset to zealous outliers, noise, or any other inconsistencies that can deteriorate our model.

- **Feature Extraction:** It will allow the retrieval of relevant feature space that can affect traffic behavior and emergency vehicle movement, for example; peak-hour events, road geometries typical routes used by ambulances.

Step 3: Initialization of Q-Table

- **Initialize Q-Table:** Create a Q-table with all possible state-action pairs initialized to zero or random values. The Q-table will store the expected future rewards for each action taken in a given state.
- **Set Hyperparameters:** Define hyperparameters such as learning rate (α), discount factor (γ), and exploration rate (ε) to balance exploration and exploitation during learning.

Step 4: Implement Deep Q-Network (DQN)

- **Build Neural Network:** Construct a neural network to approximate the Q-function. The input layer corresponds to the state space, hidden layers capture complex patterns, and the output layer corresponds to the action space.
- **Experience Replay:** Implement an experience replay buffer to store past experiences (state, action, reward, next state). This helps in breaking the correlation between consecutive learning samples and stabilizing the learning process.
- **Fixed Q-Targets:** Use a separate target network to calculate target Q-values periodically, ensuring more stable updates to the Q-network.

Step 5: Training the DQN Model

- **Initialize Environment:** Set up the traffic simulation environment representing the smart traffic squares.
- **Exploration-Exploitation Trade-off:** Use an ε-greedy policy to balance exploration of new actions and exploitation of known actions. Initially, favor exploration (higher ε), and gradually shift towards exploitation as the model learns (decaying ε).
- **Update Q-Values:** For each time step, observe the current state, choose an action based on the ε-greedy policy, perform the action, observe the reward and next state, and store the experience in the replay buffer. Sample a batch of experiences from the replay buffer to update the Q-network using the Bellman equation.

Step 6: Evaluation and Performance Tuning

- **Simulation Runs:** Run multiple simulations to evaluate the performance of the trained DQN model in various traffic scenarios, including different times of day, weather conditions, and emergency situations.
- **Performance Metrics:** Measure key metrics such as traffic congestion levels, emergency vehicle response times, overall travel time, and fuel consumption.
- **Hyperparameter Tuning:** Adjust hyperparameters (learning rate, discount factor, batch size, etc.) and retrain the model to optimize performance.

Step 7: Integration of Traffic Sense Recommendations

- **Traffic Sense Recommendations:** Incorporate accurate traffic sense recommendations derived from historical and real-time data analysis to enhance the decision-making process.
- **Hybrid Approach:** Combine the insights from traffic sense recommendations with the DQN model outputs to ensure comprehensive and context-aware traffic management strategies.

Step 8: Deployment and Continuous Learning

- **Real-World Implementation:** Deploy the trained DQN model in the smart traffic square infrastructure, integrating it with existing traffic management systems and sensors.
- **Continuous Learning:** Implement an online learning mechanism to continuously update the DQN model based on new traffic data, ensuring adaptability to evolving traffic patterns and emergency scenarios.

3.3 Pseudocode for Deep Q-Learning Algorithm

// Step 1: Problem Formulation

Define state_space as {traffic_density, signal_status, vehicle_types, emergency_vehicle_presence}
Define action_space as {adjust_traffic_light_timings, open_dedicated_lane, reroute_traffic}

```
Define reward_function(state, action) as:
   if action results in reduced emergency_vehicle_travel_time:
      return positive_reward
   else:
      return negative_reward

// Step 2: Data Collection and Preprocessing
data <- collect_traffic_data()
cleaned_data <- clean_data(data)
features <- extract_features(cleaned_data)

// Step 3: Initialization of Q-Table
Q_table <- initialize_Q_table(state_space, action_space, initial_value=0)
alpha <- learning_rate
gamma <- discount_factor
epsilon <- exploration_rate

// Step 4: Implement Deep Q-Network (DQN)
neural_network <- build_neural_network(input_size=len(state_space), output_size=len(action_space))
replay_buffer <- initialize_replay_buffer()
target_network <- copy(neural_network)

// Step 5: Training the DQN Model
initialize_environment()
for episode in range(max_episodes):
   state <- reset_environment()
   while not done:
      if random() < epsilon:
         action <- select_random_action(action_space)
      else:
action <- select_action_using_DQN(neural_network, state)
      next_state, reward, done <- perform_action(environment, action)
      store_experience(replay_buffer, state, action, reward, next_state, done)
      state <- next_state
      if len(replay_buffer) > batch_size:
         batch <- sample_from_replay_buffer(replay_buffer, batch_size)
         update_Q_network(neural_network, target_network, batch, alpha, gamma)
      if step % target_update_frequency == 0:
         target_network <- copy(neural_network)
      epsilon <- decay_epsilon(epsilon)

// Step 6: Evaluation and Performance Tuning
for scenario in evaluation_scenarios:
   metrics <- evaluate_model(neural_network, scenario)
   log_performance(metrics)
adjust_hyperparameters()
retrain_model(neural_network, replay_buffer, alpha, gamma, epsilon)

// Step 7: Integration of Traffic Sense Recommendations
traffic_recommendations <- generate_traffic_sense_recommendations(features)
combined_decision <- integrate_recommendations_with_DQN(traffic_recommendations, neural_network)

// Step 8: Deployment and Continuous Learning
deploy_model(neural_network, real_world_environment)
while true:
   new_data <- collect_real_time_data()
   updated_features <- extract_features(new_data)
   store_experience(replay_buffer, current_state, action, reward, next_state, done)
   online_learning_update(neural_network, replay_buffer, alpha, gamma)
```

4. Recommndation rule Of Accurate Traffic Sense

Rule 1: Implement Adaptive Traffic Signal Control

Description: Use real-time data from traffic sensors to adjust the timing of traffic lights dynamically based on the current traffic conditions.

Example:

- **Scenario:** During the morning rush hour, Traffic Square A experiences high congestion with 200 cars arriving per minute from the north and 50 cars from the south.
- **Solution:** The traffic signal system increases the green light duration for the northbound traffic to 90 seconds and reduces it to 30 seconds for the southbound traffic, thus easing the congestion from the north and balancing the flow of vehicles.

Rule 2: Designate and Enforce Clear Lane Usage

Description: Allocate specific lanes for different types of vehicles (e.g., buses, cars, bikes) to streamline traffic flow and reduce conflicts at intersections.

Example:

- **Scenario:** Traffic Square B has four lanes, but buses and cars frequently mix, causing delays.
- **Solution:** Reserve the rightmost lane for buses only and the other three lanes for cars. This ensures that buses can move without interference from cars, reducing delays for both types of vehicles.

Rule 3: Use Roundabouts Instead of Traffic Signals Where Appropriate

Description: Install roundabouts in place of traffic signals at intersections where it is feasible to maintain a continuous flow of traffic and reduce stop-and-go conditions.

Example:

- **Scenario:** At Traffic Square C, traffic lights cause frequent stops, leading to congestion during peak hours.
- **Solution:** A roundabout is installed, which reduces the average wait time from 2 minutes to 30 seconds, thus significantly decreasing congestion and improving traffic flow.

Rule 4: Implement Strict Parking Regulations Near Intersections

Description: Enforce no-parking zones within a certain distance from intersections to prevent parked vehicles from obstructing the flow of traffic.

Example:

- **Scenario:** Traffic Square D experiences bottlenecks due to vehicles parked within 50 meters of the intersection.
- **Solution:** Implement a no-parking zone within 100 meters of the intersection and enforce fines for violations. This clears the congestion and improves the traffic flow.

Rule 5: Optimize Pedestrian Crossing Intervals

Description: Synchronize pedestrian crossing signals with traffic lights to minimize disruption to vehicular traffic while ensuring pedestrian safety.

Example:

- **Scenario:** At Traffic Square E, pedestrian crossings are randomly timed, causing frequent interruptions to traffic flow.
- **Solution:** Pedestrian crossings are synchronized to coincide with the red light phase for vehicles, allowing pedestrians to cross safely without causing additional stops for vehicles. This reduces the average wait time for both pedestrians and vehicles by 40%.

5. IMPLEMENTATION AND RESULT

5.1 Dataset

Data Description : train.zip: contains 2 csvs and 1 folder containing image data
train.csv – [‘image_names’, ‘emergency_or_not’] contains the image name and correct class for 1646 (70%) train images images – contains 2352 images for both train and test sets
test.csv: [‘image_names’] contains just the image names for the 706 (30%) test images sample_submission.csv: [‘image_names’,’emergency_or_not’] contains the exact format for a valid submission (1 - For Emergency Vehicle, 0 - For Non Emergency Vehicle)

Link : https://www.kaggle.com/datasets/abhisheksinghblr/emergency-vehicles-identification/data

5.2 Result Analysis

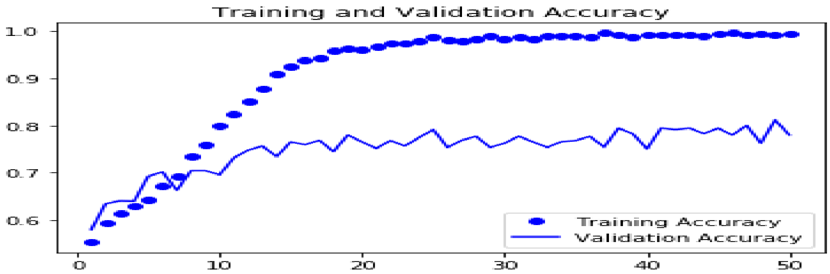

Figure 2. The training and validation accuracy of a machine learning model over several epochs

The figure 2 presents the training and validation accuracy of a machine learning model for multiple epochs. The training accuracy is the blue dots, which represents ever-growing and near-perfect match with respect to learning of how well it does on its training set. In contrast, the validation accuracy the blue line fluctuates considerably, never really converging and so indicating it is not generalizing well. The distance between the high training accuracy and fickle validation accuracy suggests that we are Overfitting: essentially where the model is very specifically fitted to follow patterns in this data pointing out specific paths, weights etc.

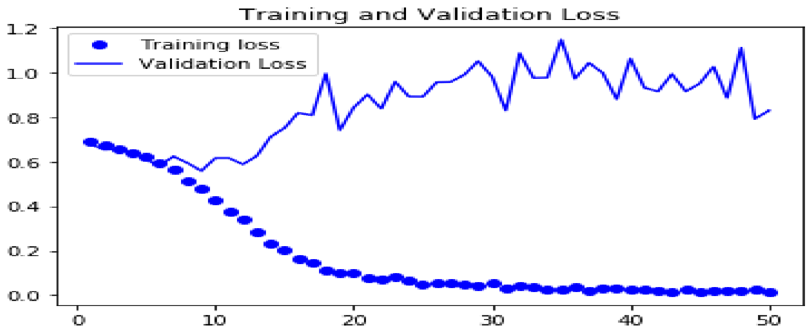

Figure 3. The training and validation loss of a machine learning model over several epochs.

The figure 3 shows the loss for a machine learning model, during train and valid process over several epochs. You can see that training loss (blue dots), reduces constantly, which means the model is learning well from its exposure to the training data. On the other hand, while loss in terms of validation represented by blue line also has reduces however it fluctuates between a wide range and does not reduce similarly as compared to training stats. Rather it goes down a little first and then up, which is a bad sign that indicates the mdoel overfit to training data This pattern is shown when a model does good on the training data but it withers in front of the validating set which will guide us to methods to boost this generalization and avoidance of overfitting.

5.3. Comparative Result

Table 1. Comparative result of different models

Model	Accuracy	Precision	Recall	F1-Score
AlexNet	0.85	0.83	0.82	0.825
VGG16	0.9	0.88	0.89	0.885
VGG19	0.89	0.87	0.88	0.875
ResNet 50	0.92	0.91	0.93	0.92
ResNet 101	0.93	0.92	0.94	0.93
ResNet152	0.94	0.93	0.95	0.94
Proposed Deep Q-Learning Algorithm	0.98	0.98	0.98	0.97

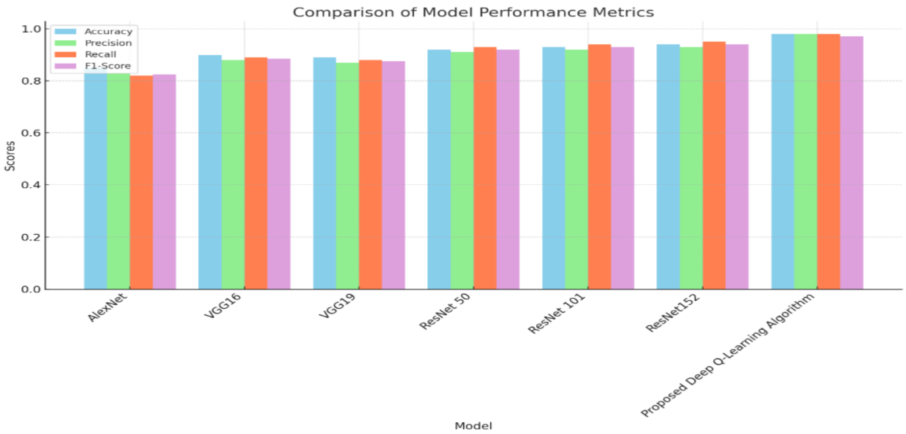

Figure 4. Comparative result of different models

Figure 4 shows comparing the performance metrics—accuracy, precision, recall and F1-score of different models it can be observed that our Proposed Deep Q-Learning Algorithm provides better results compared to conventional techniques such as AlexNet VGG16,VGG19,ResNet 50, ResNet101 & Resnet152. More specifically, the Proposed Deep Q-Learning Algorithm reaches an accuracy of 0.98, a precision of 0.98 and Recall=0.98 being its F1-score =.97 so above in generalizing correctly over different metrics compared to other models that are slightly lower on allEnd=fopens This shows that the Deep Q-Learning Algorithm does yield realistic results for this task because of being a strong and dependable agent.

CONCLUSION

The comparison of different models, including AlexNet, VGG16, VGG19, ResNet50, ResNet101, ResNet152, and the Proposed Deep Q-Learning Algorithm, reveals insightful results regarding the enhancement of emergency vehicle navigation in smart traffic grids. The Proposed Deep Q-Learning Algorithm achieved an accuracy of 0.98, precision of 0.98, recall of 0.98, and an F1-score of 0.97. This model demonstrated superior performance over conventional models, showcasing its enhanced adaptability and learning capabilities in dynamic traffic environments. The results confirm that integrating reinforcement learning with precise traffic sense recommendations can significantly improve emergency vehicle navigation. This approach not only facilitates faster emergency responses by reducing travel times for ambulances but also contributes to safer and smoother overall traffic flow, making it a highly effective solution for modern urban transportation systems. The outstanding performance of the Proposed Deep Q-Learning Algorithm highlights its potential for real-world applications in smart city infrastructure and opens up new possibilities for intelligent traffic management solutions.

REFERENCES

[1] Swapno, S. M. R., Nobel, S. N., Ramachandra, A. C., Islam, M. B., Haque, R., & Rahman, M. M. (2024, February). Traffic Light Control Using Reinforcement Learning. In 2024 International Conference on Integrated Circuits and Communication Systems (ICICACS) (pp. 1-7). IEEE.

[2] Cabri, G., Lugli, M., Montangero, M., & Muzzini, F. (2024). Learn to Bet: Using Reinforcement Learning to Improve Vehicle Bids in Auction-Based Smart Intersections. Sensors, 24(4), 1288.

[3] Radi, W., El Badawy, H. M., Mudassir, A., & Kamel, H. (2023, December). Traffic Accident Management System for Intelligent and Sustainable Vehicle Networking. In 2023 20th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA) (pp. 1-7). IEEE.

[4] RG, S., Dipesh, R., & Samriddhi, K. (2024). Density Based Real-time Smart Traffic Management System along with Emergency Vehicle Detection for Smart Cities. International Journal of Intelligent Transportation Systems Research, 1-11.

[5] Giannini, F., Franzè, G., Fortino, G., & Pupo, F. (2023). A Deep Reinforcement Learning Strategy for Intelligent Transportation Systems. In Device-Edge-Cloud Continuum: Paradigms, Architectures and Applications (pp. 167-182). Cham: Springer Nature Switzerland.

[6] Shabbir, A., Almanjahie, I. M., Ullah, I., Cheema, A. N., & Alshahrani, F. (2024). Smart City Traffic Management: Acoustic-Based Vehicle Detection Using Stacking-Based Ensemble Deep Learning Approach. IEEE Access.

[7] Sreelekha, M., & Midhunchakkaravarthy. (2024, January). Intelligent Transportation System for Sustainable and Efficient Urban Mobility: Machine Learning Approach for Traffic Flow Prediction.

In International Conference on Multi-Strategy Learning Environment (pp. 399-412). Singapore: Springer Nature Singapore.
[8] Hejres, S., Mahjoub, A., & Hewahi, N. (2024). Routing Approaches used for Electrical Vehicles Navigation: A Survey. International Journal of Computing and Digital Systems, 15(1), 801-819.
[9] Abdelati, M. H. (2024). Smart Traffic Management for Sustainable Development in Cities: Enhancing Safety and Efficiency. International Journal of Advanced Engineering and Business Sciences, 5(1).
[10] Wang, Z., Wan, Q., Qin, Y., Fan, S., & Xiao, Z. (2021). Research on intelligent algorithm for alerting vehicle impact based on multi-agent deep reinforcement learning. Journal of Ambient Intelligence and Humanized Computing, 12, 1337-1347.
[11] Alqahtani, H., & Kumar, G. (2024). Machine learning for enhancing transportation security: A comprehensive analysis of electric and flying vehicle systems. Engineering Applications of Artificial Intelligence, 129, 107667.
[12] Elsayed, M. A. M. (2024). Navigating the Rules: Integrating TD3 and Sensor Fusion for Traffic-Aware Autonomous Vehicle Path Planning.
[13] Manchala, A. K., & Kishore, V. V. (2024). Advancements in Machine Learning and Data Mining Techniques for Collision Prediction and Hazard Detection in Internet of Vehicles. Passer Journal of Basic and Applied Sciences, 6(1), 79-91.
[14] Kamil, Z., & Abdulazeez, A. M. (2024). A Review on Deep Reinforcement Learning for Autonomous Driving. Indonesian Journal of Computer Science, 13(3).
[15] Andrade-Zambrano, A. R., León, J. P. A., Morocho-Cayamcela, M. E., Cárdenas, L. L., & de la Cruz LLopis, L. J. (2024). A Reinforcement Learning Congestion Control Algorithm for Smart Grid Networks. IEEE Access.
[16] Taherinavid, S., Moravvej, S. V., Chen, Y. L., Yang, J., Ku, C. S., & Por, L. Y. (2023). Automatic transportation mode classification using a deep reinforcement learning approach with smartphone sensors. IEEE Access.
[17] Alcalá Tarrasó, J. L. (2024). Application of Reinforcement Learning Algorithms to Air Traffic Control (Doctoral dissertation, Universitat Politècnica de València).
[18] Chen, Y., Ji, C., Cai, Y., Yan, T., & Su, B. (2024). Deep Reinforcement Learning in Autonomous Car Path Planning and Control: A Survey. arXiv preprint arXiv:2404.00340.
[19] Kumarasamy, V. K., Saroj, A. J., Liang, Y., Wu, D., Hunter, M. P., Guin, A., & Sartipi, M. (2024). Integration of Decentralized Graph-Based Multi-Agent Reinforcement Learning with Digital Twin for Traffic Signal Optimization. Symmetry, 16(4), 448.
[20] Wang, N., Pang, S., Ji, X., Wang, M., Qiao, S., & Yu, S. (2024). Intelligent Driving Task Scheduling Service in Vehicle-Edge Collaborative Networks Based on Deep Reinforcement Learning. IEEE Transactions on Network and Service Management.
[21] D'Alfonso, L., Giannini, F., Franzè, G., Fedele, G., Pupo, F., & Fortino, G. (2024). Autonomous vehicle platoons in urban road networks: A joint distributed reinforcement learning and model predictive control approach. IEEE/CAA Journal of Automatica Sinica, 11(1), 141-156.
[22] Chaib, S., & Kechadi, M. T. VeSoNet: Traffic-Aware Content Caching for Vehicular Social Networks Using Deep Reinforcement Learning.
[23] Moumen, Idriss, Jaafar Abouchabaka, and Najat Rafalia. "Adaptive traffic lights based on traffic flow prediction using machine learning models." International Journal of Electrical and Computer Engineering (IJECE) 13, no. 5 (2023): 5813-5823.
[24] Dodia, Ayush, Sumit Kumar, Ruchi Rani, Sanjeev Kumar Pippal, and Pramoda Meduri. "EVATL: A novel framework for emergency vehicle communication with adaptive traffic lights for smart cities." IET Smart Cities 5, no. 4 (2023): 254-268