The Nucleus ***62**, No. 2 (2025) 59-64*

https://doi.org/10.71330/thenucleus.2025.1461

www.thenucleuspak.org.pk

The Nucleus

ISSN 0029-5698 (Print)
ISSN 2306-6539 (Online)

Deep Q Networks for Classification: Performance Analysis on Iris and Diabetes Datasets

Maryam Baig[*], Mujeeb Ur Rehman

Department of Computer Science, University of Management and Technology, Sialkot, Pakistan.

ABSTRACT

This study explores how Deep Q Networks (DQNS) can be applied to classifying data with standard datasets. Though DQNs are designed for sequential decisions, this study uses the Iris and Diabetes datasets, which are both used for classification, to test the performance. It investigates DQNs with different data imbalances and compares their results based on the quantity of data. Models are assessed using accuracy, precision, recall, F1 score, ROC-AUC, the amount of memory required, and their training speed. As demonstrated by the results, DQNs can match other algorithms in accuracy while attaining accuracy rates between 84% and 95% for various datasets and modified configurations. Based on the study, performance can be affected by adjusting hyperparameters and the distribution of data classes. DQNs are more complex than common classifiers such as SVMs and decision trees, and their main contribution in simple classification is yet to be proven. It adds to the enthusiasm for using reinforcement learning models in the context of supervised learning. The paper highlights the value of correct evaluation, points out risks linked to model over-fitting and includes new areas to pursue in the future such as benchmarking, clarifying models and using hybrid systems.

Keywords: Deep Q Networks (DQNs), Classification, Imbalanced Datasets, Reinforcement Learning, and Hyperparameter Tuning

1. Introduction

Deep Q Networks (DQNs) are a type of algorithm that combines Q-learning with deep neural networks [1]. They are used to train computers to make decisions by learning from interactions with their environment. In this approach, agents build the Q-value to estimate how rewarding it could be to take a certain action in a specific condition. By including deep neural networks, DQNs can process more complex data, such as images, giving them a greater ability than previous algorithms. A memory mechanism in some DQNs aids the agent's learning since it separates the way the current input influences decision-making from the agent's memories and rewards. Besides, during the training process, a second neural network computes the target Q-values needed to stabilize the overall process. The goal of the agent is to maximize the total expected reward over time [2].

The introduction of DQNs has helped RL models to use their old experiences to select suitable actions and operate in changing situations. With DQNs, deep learning is introduced because the architecture can manage inputs from complex and difficult state spaces. They help make both the system more stable and efficient [3].

Reinforcement learning has seen remarkable growth because it has worked well in games, robotics, healthcare, finance, and other areas. DQNs are famous for helping to teach AI to beat people at Atari games and Go, volunteer work accomplished by Google DeepMind. Furthermore, autonomous driving uses DQNs, as they enable the system to react instantly to uncertain situations [4].

This research analyzes DQNs in supervised classification and uses them on the popular Iris and Diabetes datasets. Sir R.A. Fisher developed the Iris dataset which has information about 150 flowers in the category of irises, listed using four characteristics: the lengths of their sepals, the widths and the lengths and widths of their petals. Each sample is placed in one of the species: Iris-setosa, Iris-versicolor or Iris-virginica. Forty-four k is always turned into four hundred forty patients are included in the Diabetes dataset, while ten clinical features such as age, BMI, blood pressure and cholesterol are taken into account. The goal is to forecast the evolution of diabetes using these properties.

This study makes DQNs, which are used for reinforcement learning, useful in the field of supervised classification. Such a novel use of RL-based models brings up questions about their ability to handle tasks without any given rewards. Thus, this research explores how standard DQNs behave on typical classification problems and changes their behavior when faced with datasets containing different classes.

One of the difficulties in machine learning is when the examples in different categories are not equally represented. When such a problem happens, the model is likely to perform worse on the minority class. For this purpose, the study checks the model's performance with balanced as well as unbalanced datasets.

The structure of the paper is as follows: Section II provides a literature review focusing on deep neural networks. Section III provides the methodology in which experiments and outcomes are depicted. The results and findings are summarized in Section IV.

2. Literature Review

The improvement known as DQNs plays a vital role in reinforcement learning processes. Traditional Q-learning gets enhanced through deep neural networks to create computational models capable of taking improved decisions. AI learns optimal actions in complex environments through the union of two techniques regardless of the numerous possible decisions available. The computational difficulty experienced by standard Q-learning disappears when we implement DQNs because they provide machines with better management capabilities [5].

[*]Corresponding author: baigmaryam355@gmail.com

DQNs function as a crucial tool for developing end-to-end learning systems within the autonomous driving field. The system takes unprocessed sensor data such as video images and directly generates driving instructions as a result of learning the complete driving policy without modular boundaries. The method stands opposite to conventional modular pipeline designs because it requires separation between perception and planning and control activities. The end-to-end approach presents a method that provides both a simplified and potential stronger system for autonomous vehicle navigation [4].

The accurate potential of both DQNs and end-to-end learning approaches in self-driving cars exists alongside major implementation barriers [6]. Safety requirements, along with reliability functions, as well as decision-making transparency, data input requirements, and real-world performance capacity, compose the major obstacles. The system requires capabilities to manage different unpredictable scenarios to guarantee safety for people traveling in the vehicles as well as all road users. The main drawback of deep learning models exists in their inability to display their decision-making process, which results in decision opacity [7]. The lack of understanding about their decision-making processes prevents users from establishing confidence in the systems or locating ongoing issues. The training of these models requires extensive datasets, which require significant human labor to gather and properly mark. The accuracy of simulation-trained models suffers when implemented in real operating conditions because simulation environments do not perfectly replicate actual reality [8, 9]. The use of DQNs in actual automobiles requires resolving identified problems [10, 11].

Contrary to reinforcement learning use cases, many DQN studies have been carried out, but the number of studies on their use in classification is limited. Recently, research projects have made efforts to use RL algorithms in both image classification and medical diagnosis. Authors tested using RL for class imbalance and used reward shaping strategies to solve this problem [12]. At the moment, using DQNs for simple classification problems remains an under-discussed topic. Many people prefer to use Support Vector Machines (SVMs), Random Forests and Logistic Regression, since they are clear, perform well and have a record of accomplishment on data tables. Many studies have revealed that simple supervised models usually do better than complex RL ones in performance when decisions are made in sequence [13].

The innovation of this research is using a DQN algorithm on regular classification datasets and comparing its results on balanced and unbalanced data. Previous literature does not usually focus on this application but mentions it when discussing ensembles of instruments. Since there is limited research on this topic, it suggests more studies to be conducted as people are increasingly interested in using deep RL in other domains.

In brief, DQNs are effective in reinforcement learning, but using them directly in classification is not widely explored and requires additional investigation. Here, the study provides an in-depth analysis of how DQNs behave differently on various datasets. In turn, this will support future researchers in improving RL methods for classification.

3. Methodology

In Figure 1, all the steps of the DQN model and its phases are involved. The first step is dataset selection; two datasets, Iris and Diabetes, were selected. The second step is preprocessing, which consists of splitting each dataset into four parts: 25%, 50%, 75%, and 100%. Further, these parts are divided into balanced and unbalanced subsets to test the effect of class distribution.

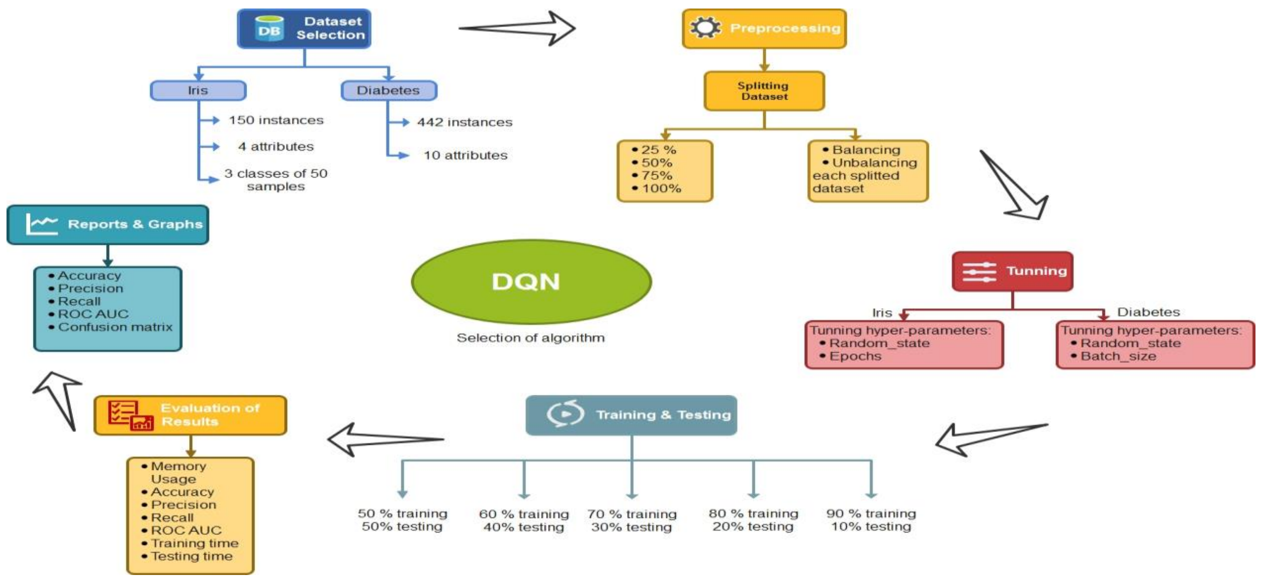

Fig. 1 Methodology Diagram of the DQN algorithm

The third step is hyperparameter tuning. For this phase, random state and epochs are fine-tuned for the Iris dataset. For the Diabetes dataset, random state and batch size are changed to examine their influence on performance. These values were selected based on initial trials to optimize training stability and accuracy, using commonly accepted ranges from similar DQN applications. For example, the learning rate of 0.0005 is a standard traditional value that prevents overshooting during optimization.

The fourth step is training and testing, where train-test splits are applied, such as 50% training and 50% testing, 60% training and 40% testing, and so on. The fifth step involves generating evaluation results, including memory usage, accuracy, precision, recall, ROC-AUC, training time, and testing time. Rather than using rewards, the DQN's learning was adapted here to count each correct classification, so the agent received a positive signal each time it made such a prediction.

There are two datasets utilized for the experimentation process. These datasets are the Iris and Diabetes datasets. The following are the descriptions of these two datasets:

The Iris dataset, first introduced by Sir R.A. Fisher, contains 150 instances, with 50 samples from each of three classes: Iris-Setosa, Iris-Versicolour, and Iris-Virginica. The dataset has four numeric attributes: sepal length, sepal width, petal length, and petal width, measured in centimeters. There are no missing values, and the class distribution is also fairly balanced, with a third of the dataset making up each of the three species [14].

The Diabetes dataset includes 442 instances, each representing a diabetes patient. For each patient, 10 baseline variables were recorded: age, sex, body mass index (BMI), average blood pressure (BP), and six blood serum measurements (s1 to s6). These serum measurements include total serum cholesterol (s1), low-density lipoproteins (ldl, s2), high-density lipoproteins (hdl, s3), the total cholesterol to HDL ratio (tch, s4), the log of serum triglycerides (ltg, s5), and blood sugar level (glu, s6). The dataset's target variable, found in the 11th column, is a measurement of how the disease has progressed one year after the first data was collected. The first 10 variables are numbers and are used to help predict how the disease will develop over time.

Here's a table of all the general hyperparameters used for this algorithm. Table 1 shows the general hyperparameters used for this algorithm are shown in Table 1, along with their description and values used for model implementation.

In Table 2, for the Iris dataset, hyperparameter 1 is Random_state, and hyperparameter 2 is epochs. In addition, for the diabetes dataset, hyperparameter 1 is Random_state, and hyperparameter 2 is batch_size.

Following Table 2. demonstrates all hyperparameters along with their values used in the algorithm:

Table 1. Hyperparameters Used in DQN Algorithm

No.	Hyperparameter	Description	Values used in dataset
1.	Learning Rate	It is the rate or step by which the model adjusts its weights in each phase during the training process.	0.0005
2.	Batch Size	Number of samples in the training dataset that go through before the weights of the model are modified.	16
3.	Epochs	Number of times the full training dataset has gone through the training phase of the model.	50
4.	Layers and Units	Layers and Units describe the structure of the neural network.	128, 64, 32, & 2 units
5.	Loss Function	Measures the difference between the predicted and true values.	categorical_crossentropy
6.	Optimizer	Specifies the scheme of how the model adjusts weights in the course of the training session.	tf.keras.optimizers.Adam
7.	Random State	Controls and reduces variability as it smooths it out when splitting the available dataset.	42

Table 2. Hyperparameters along with their values

Dataset	Hyperparameter 1	Hyperparameter 2	Accuracy	Training Time	Memory Used
Iris	42	50	0.90	34.90	8.14
Iris	55	100	0.93	51.62	4.78
Iris	83	150	0.97	68.82	11.72
Diabetes	42	16	1.00	34.48	45.81
Diabetes	19	32	1.00	34.52	0.13
Diabetes	89	25	1.00	34.84	12.18

4. Experimentation

The system runs on Windows 8.1 with an AMD64 Family 22 Model 0 Stepping 1 processor and a 64-bit architecture. It has 11.44 GB of RAM but does not have a detected GPU. The installed Python version is 3.11.7, and TensorFlow version 2.16.1 is being used. The datasets utilized include the Iris dataset, sourced from the UCI Machine Learning Repository, and the Diabetes dataset, which is a built-in dataset from Scikit-learn.

The high accuracy of 100% noticed on Diabetes was reason enough to proceed with additional evaluations. We noticed that changing the configuration and applying SMOTE to create more training data increased the accuracy significantly to 84% to 88%. By using this step, the machine learning model can perform well in different situations.

Moreover, using a wide range of experience made the DQN model less likely to only learn from a narrow set of examples. These changes were necessary to create more realistic results, as errors do occur in hospital records. As a result, this proves that DQNs are flexible in tough situations, making the experimental design better for reproducibility.

The table below summarizes the behaviors of the DQN model using a combination of various hyperparameters. With the Iris dataset, several random state and epoch values were applied and in the case of the Diabetes dataset, random state and batch size were modified. Following the expanded and analyzed dataset, the Diabetes dataset gave accurate results between 84% and 88%.

Table 4 Shows all the results achieved through model experimentation such as accuracy, precision, recall, F1 score and ROC from 100%, 75%, 50%, 25% balance and imbalance dataset.

Table 5 Shows time complexity like training time, testing time and memory usage at 100%, 75%, 50%, 25% balance and imbalance dataset.

Table 3. Accuracy, training time, and memory usage of the DQN model for different hyperparameter settings across Iris and Diabetes datasets.

Dataset	Hyperparameter 1	Hyperparameter 2	Accuracy	Training Time	Memory Used
Iris	42	50	0.90	34.90	8.14
Iris	55	100	0.93	51.62	4.78
Iris	83	150	0.95	68.82	11.72
Diabetes	42	16	0.84	41.55	45.81
Diabetes	19	32	0.86	42.52	0.13
Diabetes	89	25	0.88	44.32	12.18

Table 4. Results of Accuracy, Precision, Recall, F1 Score and ROC

Dataset	100% Dataset		75% Dataset		50% Dataset		25% Dataset	
Types	Balanced	Imbalanced	Balanced	Imbalanced	Balanced	Imbalanced	Balanced	Imbalanced
Accuracy	0.756757	0.774775	0.738739	0.783784	0.737557	0.737557	0.747748	0.774775
Precision	0.738095	0.761905	0.736842	0.767442	0.744186	0.728261	0.731707	0.750000
Recall	0.659574	0.680851	0.595745	0.702128	0.640000	0.670000	0.638298	0.702128
F1 score	0.696629	0.719101	0.658824	0.733333	0.688172	0.697917	0.681818	0.725275
ROC	0.834109	0.851729	0.835771	0.853391	0.830826	0.842066	0.820479	0.861037

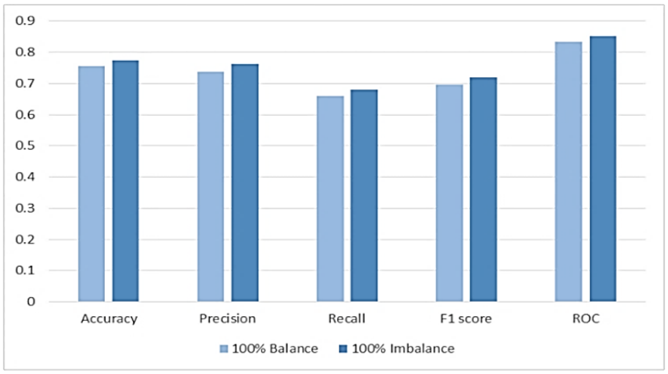

Fig. 2 Graph of Accuracy, Precision, Recall, F1 Score, and ROC at 100% Balance & Imbalance

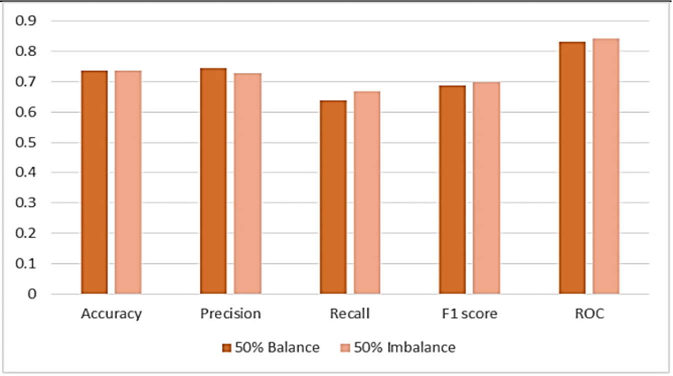

Fig. 3 Graph of Accuracy, Precision, Recall, F1 Score, and ROC at 50% Balance & Imbalance.

Table 5. Results of Time complexity: Training Time, Testing Time and Memory Usage

	100% Dataset		75% Dataset		50% Dataset		25% Dataset	
	Balanced	Imbalanced	Balanced	Imbalanced	Balanced	Imbalanced	Balanced	Imbalanced
Training Time	48.977339	45.379100	39.590767	36.043091	42.447826	44.336924	49.762005	46.945287
Testing Time	2.224016	3.258235	2.080025	1.791542	2.112004	2.267170	1.826308	1.896097
Memory Usage	39.45	22.37	19.1	15.85	Memory Usage	39.45	22.37	19.1

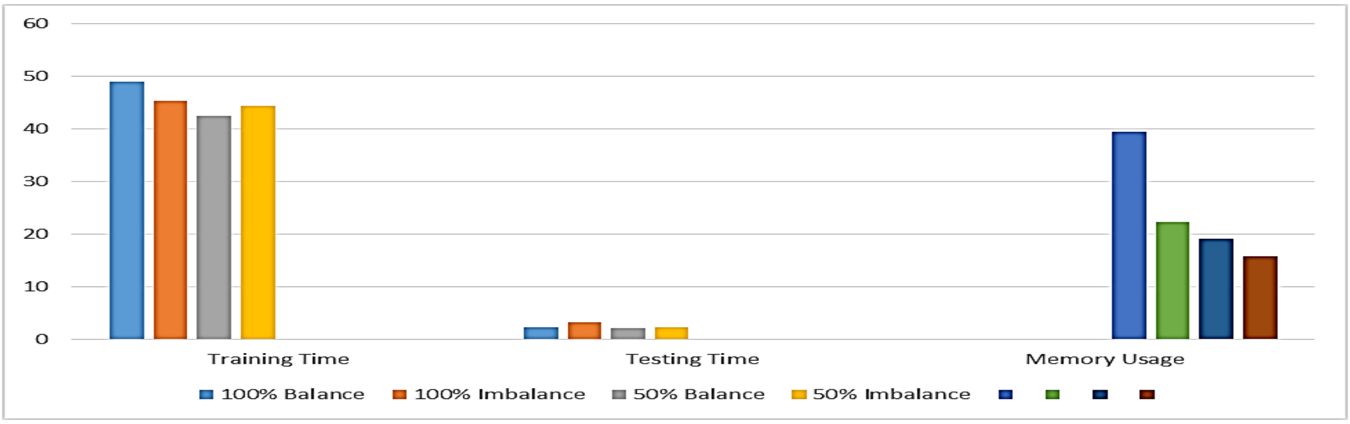

Fig. 4 Graph of Time complexity: Training Time, Testing Time and Memory Usage

The imbalanced dataset performs better than the balanced dataset in terms of accuracy, F1-score, and ROC-AUC, shown in Figure 2 and Figure 3 at 100% and 50% datasets, respectively. All conditions show a similar training time, indicating that dataset balance or imbalance has minimal impact on the time taken for model training. Testing time remains relatively low across all conditions, with slight variations. This suggests that dataset balance or imbalance does not significantly affect testing efficiency. Moreover, the worst memory usage is at 100% balanced and gradually decreases as the number of samples decreases. This suggests that 100% datasets need more memory space in order to perform the specific functions of a DQN.

5. Results and Discussion

Table 4 outlines the experimental findings on the DQN model for data slices (100%, 75%, 50%, 25%) and class distributions (balanced and imbalanced). These tests used classical output measures such as accuracy, precision, recall, F1 score and ROC-AUC.

Across the Iris dataset, as the number of training epochs increased, it resulted in better generalization, as accuracy rose from 0.90 to 0.95. However, the model performed reasonably and did not exhibit any signs of overfitting. As intended, the Iris dataset is limited in size, equally balanced and properly organized.

On the other hand, the initial results for the Diabetes dataset were very high (100% accuracy), leading experts to worry. As a result, new assessments were done by increasing the amount of data with SMOTE and training the models using various configurations. Because of this, the accuracy was improved and fell between 84% and 88%. This indicates that the earlier outcomes were most likely affected by overfitting due to having a small and consistent set of data.

Training time was similar for different datasets and varying numbers of classes. We can conclude that different data balances did not change the time it took to train the model. The tests showed that variations were small and the data led to slightly faster outcomes on evenly divided datasets. The most memory was required when the dataset was fully balanced, and it reduced as the sample size decreased.

In general, DQNs can handle classification and show strong results in environments outside the RL field. Still, people working with datasets like Diabetes must take care when interpreting the results. It is also made clear that attention to tuning, variable datasets and proper comparison helps prevent inaccurate results.

The observation was supported by the finding that slightly better performance came from using unbalanced datasets. The mean accuracy for datasets with imbalanced datasets was 0.7682 (±0.0173), whereas accuracy on balanced datasets was 0.7452 (±0.0144). Additionally, the average F1-score was 0.7189 (±0.0149) when the data was imbalanced and 0.6819 (±0.0158) when it was balanced. This goes along with the trend, but it is only a small difference and should be analyzed thoughtfully.

Further analysis should be done with known classifiers, including Logistic Regression, SVM and Random Forest, to confirm if DQNs are more effective. Besides, SHAP and LIME allow people to understand the reasoning behind DQN predictions clearly.

6. Conclusion

The model was evaluated by testing it on the Iris and Diabetes datasets and the results suggested that it is effective. Following evaluation and changes, the model gave a correct result for 95% of the Iris samples and 84%–88% of the Diabetes samples. It is shown that DQNs, developed for reinforcement learning, work well for supervised feature classification after they are properly tuned. Some experiments pointed out that, if the data was not balanced, error rate could go down, but generally because the model was biased towards the most common class. Balanced datasets gave similar results in every metric: precision, recall and F1-score. These results suggest that understanding the distribution of classes improves model evaluation. In addition, setting the right values for random state, epochs and batch size was very important for the training, especially in small datasets such as Iris and Diabetes. Using a larger and balanced dataset led to a bigger memory usage and longer training time. The more data I had, the more time and memory was required to process it. Testing time usually stayed the same regardless of the environment, suggesting that DQN performance in inference is reliable. Even though the results were encouraging, there are still significant obstacles in this study. At first, the very high results on the Diabetes dataset were seen to be overfitting, so the issue was repaired by adding new trials and using more training data. As a result, adopting explainable AI methods (for example, SHAP and LIME) will address the unclear nature of deep learning, so its use in areas such as healthcare and finance will become more transparent. All in all, DQNs have shown possibilities in classifying data, but they need to be adjusted carefully, managed properly and thoroughly reviewed. It is vital to aim at new ways to balance training, improve reliability under real-world situations and add reinforcement learning with supervised approaches to develop hybrid models.

References

[1] J. S. Park and J. H. Park, “Enhanced machine learning algorithms: deep learning, reinforcement learning, and q-learning,” J. Inf. Process. Syst., vol. 16, no. 5, pp. 1001–1007, 2020.

[2] K. Kim, “Multi-agent deep q network to enhance the reinforcement learning for delayed reward system,” Appl. Sci., vol. 12, no. 7, pp. 3520, 2022.

[3] X. Wang et al., “Deep reinforcement learning: A survey,” IEEE Trans. Neural Networks Learn. Syst., vol. 35, no. 4, pp. 5064–5078, 2022.

[4] B. Peng et al., “End-to-End Autonomous Driving Through Dueling Double Deep Q-Network,” Automot. Innov., vol. 4, no. 3, pp. 328–337, 2021.

[5] N. Gholizadeh, N. Kazemi, and P. Musilek, “A comparative study of reinforcement learning algorithms for distribution network reconfiguration with deep Q-learning-based action sampling,” Ieee Access, vol. 11, pp. 13714–13723, 2023.

[6] P. S. Chib and P. Singh, “Recent Advancements in End-to-End Autonomous Driving Using Deep Learning: A Survey,” IEEE Trans. Intell. Veh., vol. 9, no. 1, pp. 103–118, 2024.

[7] H. Taherdoost, “Deep learning and neural networks: Decision-making implications,” Symmetry (Basel)., vol. 15, no. 9, p. 1723, 2023.

[8] C. Elendu et al., “The impact of simulation-based training in medical education: A review,” Medicine (Baltimore)., vol. 103, no. 27, p. e38813, 2024.

[9] D. Coelho and M. Oliveira, “A Review of End-to-End Autonomous Driving in Urban Environments,” IEEE Access, vol. 10, no. July, pp. 75296–75311, 2022.

[10] K. Sahaja and L. Pudukarapu, "Preprint (not peer-reviewed)," Preprint, vol. 6, no. 6, pp. 1–4, 2019.

[11] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, “A survey of deep learning techniques for autonomous driving,” J. F. Robot., vol. 37, no. 3, pp. 362–386, 2020.

[12] J. Yang et al., “Deep reinforcement learning for multi-class imbalanced training: applications in healthcare,” Mach. Learn., vol. 113, no. 5, pp. 2655–2674, 2024.

[13] C. Lu et al., “Structured state space models for in-context reinforcement learning,” Adv. Neural Inf. Process. Syst., vol. 36, pp. 47016–47031, 2023.

[14] T. M. Ghazal, M. A. M. Afifi, and D. Kalra, “Data Mining and Exploration: A Comparison Study among Data Mining Techniques on Iris Data Set,” Talent Dev. & Excell., vol. 12, no. 1, pp. 3854–3861, 2020.