Hybrid Machine Learning Approach for Vehicle Routing

Group 2770
Martin MICHAUX (5701333), Nishant AKLECHA (5693357)

January 27, 2024

Abstract

The Vehicle Routing Problem (VRP) involves finding the best routes for vehicles to serve customers with varying demands, locations, and time windows. It is important for logistics and transportation to reduce costs and improve customer satisfaction.
This research proposes to combine Machine Learning architectures, including Graph Neural Network (GNN), Convolutional Neural Network (CNN), and K-Means Clustering (KMC), with Constraint Programming (CP) to enhance routing solutions for the VRP. The GNN captures spatial relationships and dependencies among locations, while the CNN generates vehicle assignment outputs based on input images. The KMC provides initial solutions, which are refined using CP. Experimental results, especially for the KMC, demonstrate the effectiveness of our approach in improving objective performance compared to the baseline CP solver.
Keywords: Vehicle Routing Problem, Machine Learning, Graph Neural Network, Convolutional Neural Network, K-Means Clustering, Constraint Programming

1 Introduction

The Vehicle Routing Problem (VRP) is a challenging combinatorial optimization problem that involves finding the optimal routing plan for a fleet of vehicles to serve a set of customers with varying demands, locations, and time windows. The problem is highly relevant to logistics and transportation industries, where efficient routing plans are crucial for reducing operational costs and improving customer satisfaction [3].

To tackle the VRP, our research proposes a novel approach that combines various Deep Learning (DL) and Machine Learning (ML) state-of-the-art architectures and constraint programming (CP) techniques. Unlike traditional non-ML approaches, ours involves training various models using partial solutions derived from the CP algorithm. These models generate output predictions, which are subsequently utilized as initial solutions for the CP algorithm employed in solving the VRP.

The proposed hybrid approach addresses several limitations of existing solvers in solving the VRP. Conventional solvers often struggle with large-scale instances of the VRP, as the problem's complexity grows exponentially with the number of customers and vehicles. This leads to lengthy computation times and suboptimal solutions. Moreover, these solvers typically rely on heuristics that lack the ability to exploit the underlying structure and patterns in the data.

In contrast, our hybrid approach leverages the power of DL and ML techniques to overcome these limitations. By training DL and ML models on partial solutions obtained from a CP algorithm, we can harness the predictive capabilities of ML to generate improved initial solutions. This integration of ML and CP enables a more efficient and effective search process, significantly reducing the computation time required to find high-quality solutions.

Our implemented ML models encompass both supervised and unsupervised approaches, namely the Graph Neural Network (GNN)[5], Convolutional Neural Network (CNN), and K-Means Clustering (KMC) [1]. These models exploit the inherent structure in location coordinates. The supervised models, specifically the GNN and CNN, utilize the generated partial solutions to construct a network based on location data, taking into account vehicle assignments and capturing underlying routes through images, respectively. On the other hand, the unsupervised KMC learns from the structural patterns in the locations, without considering

assignments.

Our research makes the key contribution of exploring the integration of ML techniques based on CP partial solutions and search techniques. This novel approach aims to leverage ML's predictive capabilities to generate improved initial solutions for the VRP. Overall, our research offers insights into the practical application of ML and search techniques for enhancing the efficiency and effectiveness of solving complex optimization problems like the VRP.

In this research, the main result highlights the significant performance improvement achieved by incorporating the KMC technique into the CP solver for the VRP. For larger problem instances with 500 locations and 20 vehicles, the KMC integration led to an impressive enhancement of 130% compared to the baseline CP solver. Furthermore, even for smaller instances involving 50 locations, the KMC demonstrated its effectiveness by delivering an average improvement of 75%. These results underscore the potential of leveraging KMC as a valuable tool in optimizing routing solutions for the VRP, showcasing its ability to yield superior performance across different problem scales. However, it is important to note that the deep learning models, the GNN and CNN, did not reach significant performance levels. This outcome is attributed to the limited availability of training data, which hindered the models' ability to effectively learn and generalize from the provided information. As a result, the integration of these deep learning models did not lead to a notable enhancement of the overall performance of the CP solver.

This paper begins with a background (Section 2) providing necessary context, followed by a review of related work in 3. The main body of the report consists of technical sections (Section 4) presenting our approach, with the experimental results and discussions (Section 5). Finally, it is concluded with the main findings and future directions in Section 6.

1.1 Research Objectives

The main objectives of our research are:

- Proposing novel approaches that integrate ML techniques to improve the performance of the VRP.
- Evaluating the effectiveness of these approaches through comprehensive experiments and performance metrics.
- Comparing the results with baseline methods to assess the impact and benefits of our proposed ML-assisted CP approach.

By addressing these objectives, we aim to enhance the efficiency and effectiveness of vehicle routing solutions, particularly in applications such as delivery services, where optimizing routes can significantly impact operational costs and customer satisfaction.

2 Background

In our implementation, we consider the standard variant of the VRP. Our goal is to determine the optimal allocation of vehicles to minimize total travel costs. In this section, we provide a formal overview of this VRP variant and its key components through the following problem formulation:

- **Number of Vehicles**: A predefined number of vehicles available for serving customer demands. The goal is to determine the optimal allocation of these vehicles to minimize total travel costs.
- **Customer Locations**: A set of customer demands, each associated with a specific location represented by the coordinates (x, y) in a 2D space.
- **Depot**: The central location from which the vehicles start and end their routes. It serves as the base for the vehicle operations.
- **Grid Size**: The VRP takes place within a 2D grid, which encompasses all the customer locations and the depot. The dimensions of this grid determine the spatial context of the problem.
- **Maximum Distance**: Each vehicle in the VRP has a predefined maximum travel distance it can cover within a single route. This constraint ensures that the vehicles operate within practical limitations.

With that, as CP solver, we decided to use **OR-Tools**. OR-Tools is an open-source software library developed by Google[1] that provides a wide

[1] https://developers.google.com/optimization/routing/vrp

range of optimization algorithms and tools. It includes various solvers and algorithms for combinatorial optimization problems, including the VRP.

3 Related work

In this section, we aim to highlight the key differences and resemblances between our project and some notable works in the field.

In [4], SeaPearl is presented as a solver that employs reinforcement learning for decision-making in the search process. While SeaPearl emphasizes flexibility and ease of use, it does not match the performance of existing solvers. In comparison, our research focuses on leveraging the complementary strengths of ML and CP to improve the efficiency and effectiveness of the search process, aiming for better routing solutions.

Another notable work is introduced in [2] where the authors investigate the use of ML to enhance the efficiency of solving VRPs with time windows. The study trains models to mimic the decisions of an expert solver, improving route predictions for a state-of-the-art solver. In contrast to this approach, our research focuses on the integration of ML and CP. By combining their strengths, our objective is to improve the overall efficiency and effectiveness of the search process, ultimately leading to better routing solutions.

Additionally, a more related paper is [12]. In contrast to this work, our approach differentiates itself by providing hints to the CP algorithm. While their work focuses on neural networks for local search and crossover operations, we specifically explore the integration of neural networks to offer suggestions and hints to enhance the CP algorithm in the VRP.

Finally, by training our ML models on the solutions generated by a CP algorithm rather than relying on a direct ML solution, our approach differentiates itself from [6]. Their paper presents a new approach specifically designed for solving the VRP with hard time windows, with a focus on its application in a supermarket chain. However, they did not consider training their ML model on the partial solutions from the CP solver.

These papers contribute to the exploration of integrating ML and search techniques for solving complex optimization problems like the VRP. However, our research distinguishes itself by combining the CP partial solutions and the ML models to enhance the search process of the CP and provide novel insights into improving routing solutions.

4 Methodology

The methodology section details the steps taken to set up, conduct experiments, and measure the effectiveness of our CP assisted by ML approach.

We describe the configuration and implementation of our proposed approach, including the integration of ML techniques into the CP framework. The experiments are conducted using a diverse set of problem instances, allowing for a comprehensive evaluation.

4.1 Hypothesis

We hypothesize that utilizing a high-quality initial solution, obtained by running the CP solver for a defined duration and capturing the partial solution generated, will result in a significant enhancement in the performance of the CP algorithm for VRP.

Specifically, we expect the objective scores, particularly the total distance travelled, to be significantly lower when utilizing the partial solution as the initial solution compared to running the CP algorithm without it.

4.2 Validating the Hypothesis

We created multiple VRP instances and performed the following experiment. For each problem instance, we executed the CP solver for two consecutive runs. In the first run, the solver was given a random initial solution, while in the second run, the previous solution obtained from the first run was used as the initial solution, as seen in Figure 1.

To ensure a fair comparison, we set a time limit of 10 seconds for each run. We found that during the second run, the solver was able to provide a better solution every time.

Overall, we observed an improvement of 30%-50% for our objective value, depending on the number of locations and the specific

problem instance.

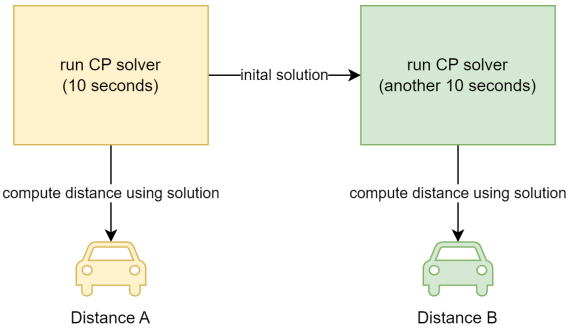

Figure 1: Validating the Hypothesis

4.3 Baseline Results

To establish a basis for comparison with our proposed approach, we implemented a baseline using a CP solver with a fixed runtime of 10 seconds. This choice reflects real-world scenarios commonly encountered in applications such as delivery services, where longer computational times. For example, a wait time of one hour is not practical for food delivery or taxi apps.

4.4 Methodology of Experiments

The methodology of our experiments involved conducting various unsupervised and supervised ML methods to obtain partial solutions for the VRP. These ML methods were employed as a preliminary step to generate initial solutions. The time taken by these ML methods to produce the partial solutions was denoted as X. Subsequently, we allocated the remaining time $(10-X)$ to the CP solver to refine and optimize the solutions further. Our evaluation focused on measuring the improvement in objective value and the distance travelled by vehicles in the VRP, as seen in Figure 2.

Furthermore, we compared the performance of our approach to the baseline CP solver with a fixed runtime of 10 seconds to assess the improvements achieved by incorporating ML methods.

By comparing the objective values and distance travelled for the initial ML-based partial solutions with the refined solutions obtained using the CP solver, we assessed the effectiveness of our approach. The improvement in these measures provided insights into the impact of combining ML methods with the CP solver in addressing the VRP. The experiments were conducted systematically and repeated multiple times to ensure reliable and statistically significant results.

This methodology allowed us to quantitatively evaluate the performance and effectiveness of our proposed approach in solving the VRP. The results obtained from these experiments serve as evidence of the value added by incorporating ML techniques into the optimization process and provide valuable insights for potential applications in real-world scenarios.

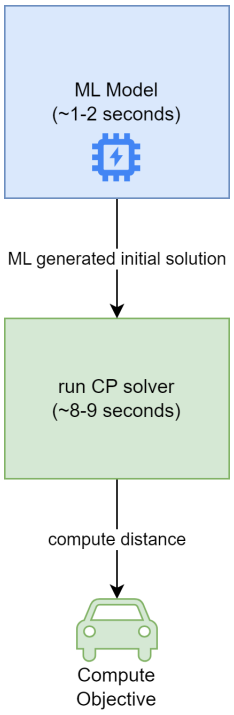

Figure 2: Methodology of Experiments

4.5 Dataset Preparation

The availability of a dataset is crucial for training ML models, particularly supervised learning approaches. A dataset provides the necessary input-output pairs that enable the models to learn patterns and make accurate predictions. In our research, we generated a dataset consisting of 1500 instances of the VRP. Each instance comprised 50 random locations within a 100×100 grid, with a fixed number of

5 vehicles assigned to the locations and a fixed location of the depot in the centre of the grid. We were only able to generate 1500 instances due to time constraints faced for our study.

Fixing the number of vehicles and the depot location allowed the models to learn the relationship between the number of vehicles and the routing optimization process. Although these fixed numbers may not precisely reflect real-life scenarios where the number of vehicles and depot location vary, it represents a simplified yet relevant approximation frequently encountered in practical applications.

Statistic	**Value**
Number of Instances	1500
Grid Size	100×100
Solver Runtime	100 seconds
Locations/instance	50
Vehicles/instance	5
Avg Objective Value	20559

Table 1: Dataset Statistics

5 Experiments

In this section, we delve into a series of experiments aimed at validating the effectiveness of our proposed approach. We commence by offering an overview of the experimental setup, including the datasets employed. Subsequently, we explore our three key components: KMC, GNN, and CNN, detailing their problem instance, architecture, methodology, encountered challenges, limitations, and ultimately, the obtained results. By comprehensively analyzing these aspects, we gain valuable insights into the performance and potential of our approach.

5.1 K-Means Clustering

5.1.1 Problem Instance and Visualization

The experiment conducted in this study draws inspiration from the Sweep Algorithms (SA) mentioned in Reference [1]. In our research, we built upon the principles of the VSA to develop a novel optimization algorithm for addressing a specific variant of the VRP. By incorporating key elements from the VSA, such as the sweep procedure and adaptive routing strategies, we aimed to improve the efficiency and effectiveness of our algorithm.

Additionally, Figure 4 also includes a comparison with the baseline CP solver, which is granted a fixed runtime of 10 seconds. This allows for a visual comparison between our approach and the baseline CP solution. An example problem instance can be visualized in Figure 3.

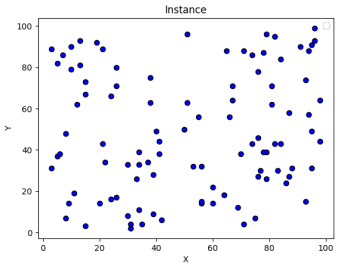

Figure 3: Example problem instance with 100 locations on a 100*100 grid.

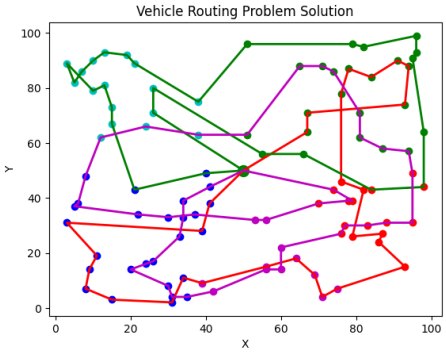

Figure 4: Baseline solution after running the CP solver for 10 seconds (Objective value: 52444)

5.1.2 Angle-Based K-Means Clustering

In this step, we perform K-means clustering based on their location and angle to the depot to generate initial solutions. The normalized angles are obtained using the code snippet:

$$normalized\_angle = (angle + 2\pi) \bmod 2\pi$$

The clustering is applied around the depot location, situated at [50,50], with the number of clusters $k = 5$, corresponding to the number of vehicles. The resulting clusters are visualized

in Figure 5, with each cluster represented by a distinct colour.

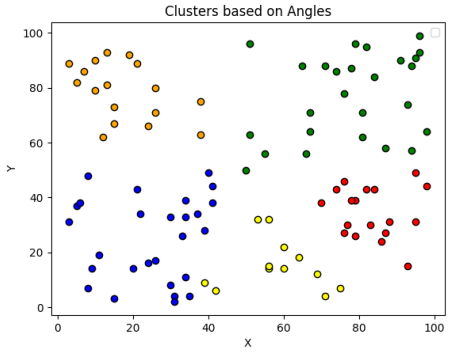

Figure 5: K-means clusters using normalized angles and k=5.

5.1.3 Challenges with Point Order and Solution Improvement

Directly utilizing the clustering solution poses a challenge, as it results in scrambled point orders, leading to a significant increase in the objective function, as shown in Figure 6.

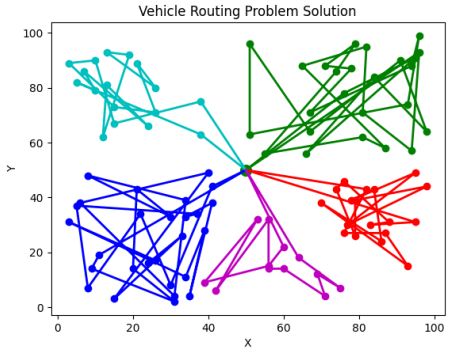

Figure 6: Solution using clusters as the initial solution (Objective value: 90,321).

To address this issue, we employ a strategy to efficiently reorder the points within each cluster. We utilize a partial solution of the Traveling Salesperson Problem (TSP) on each cluster, executing it approximately every second for all five vehicles. The resulting subgraphs, corresponding to each vehicle's route, are shown in Figure 7.

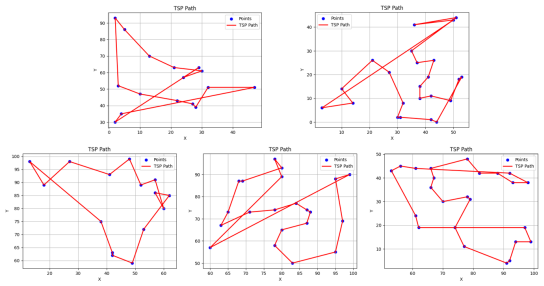

Figure 7: TSP for each expected cluster individually.

5.1.4 Refined Solution and Performance Comparison

Using the TSP solutions of the individual cluster we already get a solution that has a lower objective value than the baseline (Figure 8). This solution was achieved in ∼2 seconds, which gives us ∼8 seconds to improve the result using the CP solver.

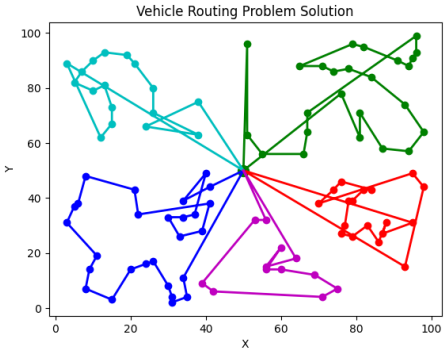

Figure 8: TSP solution in ∼2 seconds (Objective value: 42986).

The TSP-based solution, obtained by reordering the points within each cluster, is provided as input to the next stage. This refinement process yields the improved solution depicted in Figure 9, achieving the best objective function value within approximately 10 seconds.

Finally, we compare this approach to the baseline CP solver, which is granted a fixed runtime of 10 seconds. Our proposed method demonstrates a remarkable performance improvement of 100%-120% on average when compared to the baseline CP solver solution.

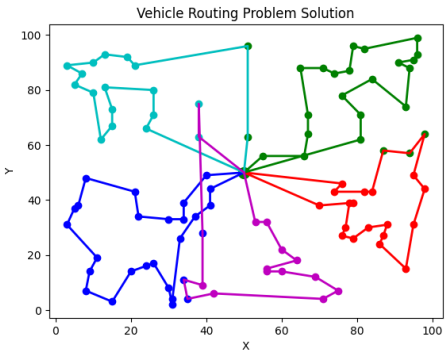

Figure 9: CP solution after TSP, after approximately 10 seconds (Objective value: 26628).

5.1.5 K-Means Experiment Results

The results presented in Table 2 demonstrate significant improvements in objective performance across various instance sizes. For VRP instances with 5 vehicles, our combined ML and CP approach achieved an average objective improvement of 100%, indicating enhanced efficiency in route planning. Furthermore, as the instance size and the number of vehicles increased, our approach consistently outperformed the baseline CP solver, resulting in objective improvements ranging from 70% to 130%. These findings underscore the effectiveness of leveraging ML techniques in conjunction with CP to optimize VRPs, regardless of the scale and complexity of the instances.

Instance Size	Improvement
50 locations, 5 vehicles	+80%
100 locations, 5 vehicles	+100%
500 locations, 5 vehicles	+120%
50 locations, 20 vehicles	+70%
100 locations, 20 vehicles	+80%
500 locations, 20 vehicles	+130%
1000 locations, 100 vehicles	+90%

Table 2: Average improvement of the objective function for varying instance sizes.

5.2 Graph Neural Network

The goal of the Graph Neural Network (GNN) is to predict the assignment of each location to a specific vehicle in the VRP. This supervised model utilizes the generated data described in Section 4.5.

5.2.1 Problem Instance and Visualization

In order to train this supervised model, we utilized the generated data, where each instance[2] consists of a collection of locations assigned to vehicles. To provide a visual representation of this data, we present an example instance in Figure 10, where each colour represents an assignment of a vehicle to a location.

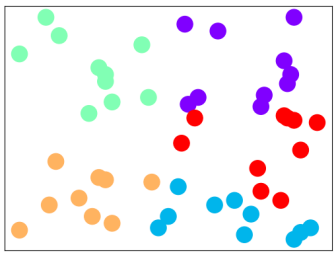

Figure 10: GNN instance example

Hence, the objective of this classification model is to predict the assignment of each location to a specific vehicle.

5.2.2 K-Nearest Neighbours Abstract Graph

To construct abstract graphs for each instance in implementing the GNN, we employed the K-Nearest Neighbours (KNN) algorithm. By selecting the k nearest neighbours using the Euclidean distance metric, we were able to capture local connectivity and spatial relationships. In our experiments, setting $k = 10$ consistently yielded favorable outcomes. In this context, "favorable" refers to achieving desirable results, such as improved model performance, accurate predictions, or better representation of the underlying structure of the VRP instances. It signifies that setting k to 10 led to positive and promising outcomes in terms of the GNN's ability to understand and model the VRP instances. However, it is important to acknowledge that this approach may have limitations in capturing more complex global patterns, and the optimal value of k may vary depending on different scenarios. Nonetheless, utilizing the KNN algorithm and abstract graphs contributes to enhancing the accuracy and meaningfulness of predictions in the GNN-based method for solving

[2]Reminder: each instance represents 50 random locations assigned to 5 vehicles in a 100×100 grid solved by the CP for 100s.

the VRP.

We conducted experiments to examine the impact of modifying the value of k on the sparsity of the instances. As shown in Figure 11, we observed a linear relationship between k and sparsity. Therefore, to determine the optimal value of k, we performed additional experiments and evaluated the final accuracy of the model. Through this analysis, we identified that the tuned value of k that yielded the best results was 10. From figure 12, you can see an example of such an abstract graph with k=10.

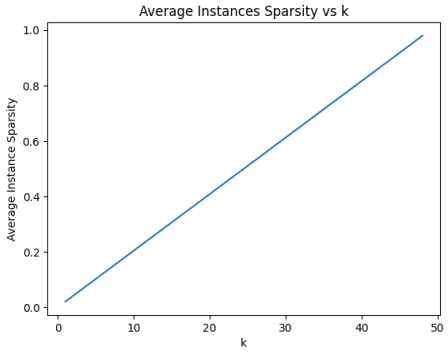

Figure 11: Average instance sparsity based on k

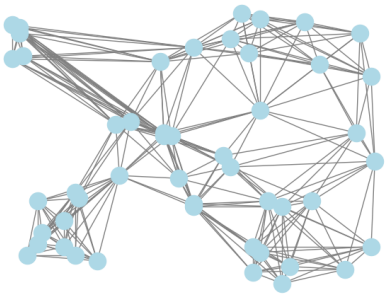

Figure 12: Abstract graph instance example with k=10

5.2.3 Architecture

The architecture of the GNN is relatively straightforward. It accepts an instance as input, which represents the coordinates of the locations, and produces an output that indicates the confidence level, as the probability, of each location being assigned to each class. In this context, each class corresponds to a different vehicle.

It consists of multiple Graph Convolutional Network (GCN) layers that are specifically designed for graph-structured data. In other words, they operate by aggregating information from neighbouring nodes in the abstract graph to compute the representation of each node. This allows the GNN to capture the relational information and dependencies among the nodes in the graph.

The GNN model can be beneficial for addressing the VRP. By utilizing the KNN abstract graph, the GCN layers enable the GNN to capture spatial relationships and dependencies among locations. This allows the model to understand the context and effectively handle variable problem sizes. Additionally, the multiple GCN layers facilitate the learning of complex patterns, resulting in improved predictions for vehicle assignments. The integration of GCN layers enhances the GNN's performance in solving the VRP by leveraging spatial relationships, accommodating varying problem sizes, and capturing intricate patterns in the KNN abstract graph. The diagram of this network is displayed in figure 13.

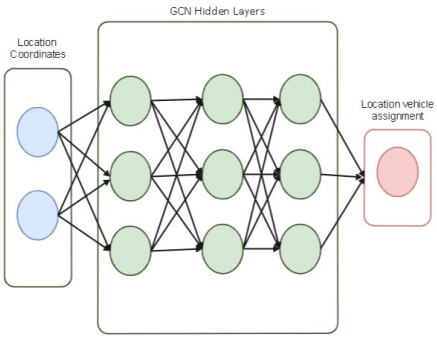

Figure 13: GNN Diagram

5.2.4 Results

We incorporated two hidden GCN layers with a size of 8 into our GNN architecture and conducted training for 200 epochs. The training loss and testing accuracy throughout the training process can be visualized in Figure 14, providing insights into the model's performance.

The plots reveal noticeable trends: the decreasing training loss indicates that the model is learning, while the relatively lower testing accuracy suggests that further improvements are required. Hence, there is room for enhancing

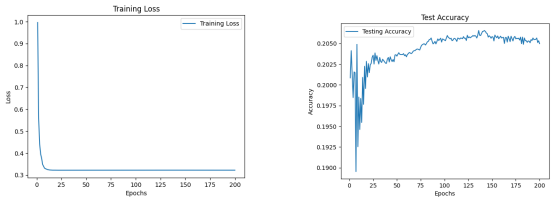

Figure 14: GNN training loss (left) and testing accuracy (right) results for 200 epochs

the model's performance.

In order to clearly see what outputs the model, we also decided to plot a predicted instance and its true value as follows in figure 15.

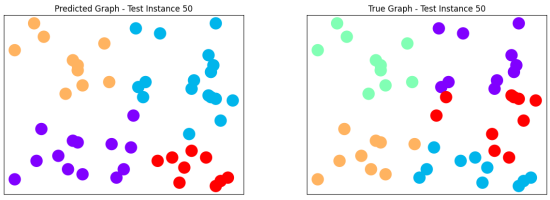

Figure 15: GNN predicted (left) and true (right) instance example result

However, we were unable to demonstrate the efficacy of the GNN in improving the performance of the CP. In each VRP run tested, the CP using the initial solution generated by the GNN model yielded similar results to the regular unguided CP. We attribute this outcome to factors discussed in the following section.

5.2.5 Limitations

This model presents several challenges that need to be addressed. These challenges encompass limited training data available for training deep learning models, the necessity for optimization techniques, and the suboptimal nature of classification for solving this specific problem.

Therefore, it is anticipated that the performance of the model can be significantly improved by increasing the available training data. Additionally, optimizing the model's architecture, exploring different GNN architectures, and conducting thorough hyperparameter tuning can lead to further enhancements.

However, while the GNN is primarily designed for classification tasks, the nature of the VRP aligns more closely with a community detection problem rather than a traditional classification problem. Therefore, one of the key challenges lies in adapting the GNN's current loss function to effectively learn the underlying communities instead of focusing solely on individual classes.

The loss function of our GNN is the Cross Entropy, to adapt it for community detection, we could explore alternative approaches that consider the structural properties of the graph and the relationships between nodes. For example, one potential modification could involve incorporating a community detection-specific objective, such as modularity optimization[11] or a variant of the normalized mutual information[10]. These metrics assess the quality of community assignments by evaluating the density of connections within communities compared to those between communities.

5.3 Convolutional Neural Network

5.3.1 Autoencoder Architecture and Input-Output Transformation

To explore an alternative approach, we implemented a Convolutional Neural Network (CNN) as an autoencoder. The purpose was to generate vehicle assignment outputs based on the input image of size 100x100x1, as described in Section 4.5. The CNN autoencoder was designed to produce an output image of shape 100x100x5, encoding the vehicle assignments using one-hot encodings.

5.3.2 Challenges and Limitations

During the experimentation with the CNN autoencoder, we encountered several challenges. One major limitation was the scarcity of available input instances. Due to this constraint, the model struggled to capture the complex patterns and dependencies required to accurately assign vehicles to locations. Additionally, the lack of a sufficient training dataset hampered the model's ability to learn and generalize effectively.

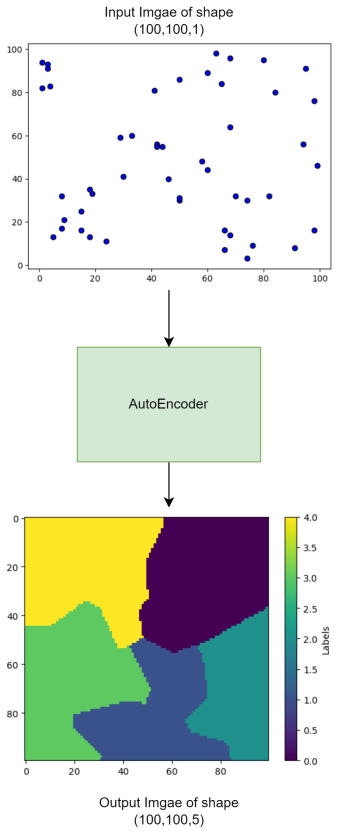

Figure 16: CNN Experiment Architecture

5.3.3 Unpromising Results and Insights

As a result of these limitations, the CNN autoencoder failed to produce promising results as shown in Figure 17 because the model produced a random output, showing that it could not learn. The generated output images did not adequately represent the desired vehicle assignments, leading to high objective values and inefficient route planning. It became evident that the limited inputs and the lack of diverse training instances were major impediments to the effectiveness of this approach.

5.3.4 Future Directions and Considerations

Based on these findings, it is clear that the CNN autoencoder approach requires a larger and more diverse dataset to capture the intricate relationships between locations and vehicle assignments. Future research should explore ways to acquire or generate a more extensive training dataset to train the CNN autoencoder effectively. Additionally, other architectural modifications and regularization techniques may be investigated to enhance the model's learning capabilities and improve the quality of the generated vehicle assignment outputs.

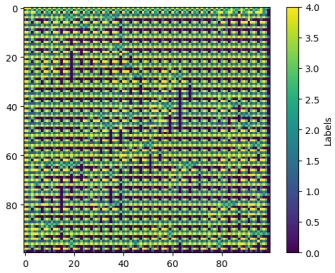

Figure 17: Output from the CNN autoencoder model

6 Conclusion

In this research paper, we explored different approaches to solving the Vehicle Routing Problem (VRP). We conducted five experiments to address the problem from various angles and evaluate their effectiveness.

6.1 Experiments

In the first experiment, we applied K-means clustering (KMC) based on normalized angles to generate initial solutions. However, we faced challenges with point order and solution improvement. To overcome this, we employed a strategy to reorder the points within each cluster using a partial solution to the Traveling Salesperson Problem (TSP). This refinement process resulted in improved solutions compared to the baseline constraint programming (CP) solver.

In the second experiment, we utilized a Graph Neural Network (GNN) model to predict the assignment of each location to a specific vehicle. We constructed an abstract graph using the K-Nearest Neighbours (KNN) algorithm and trained the GNN on the generated data. The GNN model demonstrated the ability to capture spatial relationships and dependencies among locations, leading to enhanced efficiency in route planning.

The third experiment involved implementing a Convolutional Neural Network (CNN) as an autoencoder to generate vehicle assignment outputs. However, due to the limited availability of input instances and the lack of a diverse training dataset, the CNN autoencoder approach did not produce promising results. The model struggled to capture complex patterns and dependencies, resulting in inefficient route planning.

6.2 Future Work

Based on our findings, several avenues for future research can be explored.

Concerning the KMC approach, further improvements can be made in the point reordering strategy within each cluster to enhance the solution quality. Additionally, investigating alternative clustering algorithms or integrating other optimization techniques could potentially lead to better results.

For the GNN approach, exploring different GNN architectures and hyperparameter tuning may improve the model's performance. Additionally, incorporating additional features or constraints into the GNN model could enhance its ability to handle real-world complexities.

Regarding the CNN approach, acquiring or generating a larger and more diverse training dataset is essential to improve the model's learning capabilities. Exploring different CNN architectures, regularization techniques, and data augmentation methods may also yield better results.

In our future work, we propose exploring the implementation of a Deep Reinforcement Learning (DRL) approach to further improve the solution for the VRP. This involves training an agent using the VRP CP algorithm for a short duration and leveraging the obtained partial solution to re-run the algorithm. The agent learns from the reward signal, which represents the performance difference between the two runs. This iterative process is repeated until the agent acquires a proficient policy. This approach aligns with related studies that have successfully combined DRL with dynamic programming algorithms for solving VRPs. Notably, papers such as [8] and [7] introduce Deep Policy Dynamic Programming (DPDP), which utilizes a deep neural network to derive a policy that guides and restricts the dynamic programming state space. DPDP demonstrates competitive performance, outperforming many other "neural approaches" and even strong alternatives like LKH, in solving routing problems. These findings highlight the potential of integrating deep learning techniques with dynamic programming to significantly enhance the efficiency and effectiveness of VRP solutions.

In the future, we propose performing the experiments on real-world datasets [9] and comparing the results with our simulated VRP instances.

In conclusion, our research sheds light on different approaches to tackle the VRP and highlights the importance of data availability, model architecture, and optimization techniques. The findings presented here provide valuable insights and lay the foundation for future advancements in this field.

References

[1] M. A. H. Akhand, Zahrul Jannat Peya, Tanzima Sultana, and M. M. Hafizur Rahman. Solving capacitated vehicle routing problem using variant sweep and swarm intelligence. *Journal of Applied Science and Engineering*, 20:511–524, December 2017.

[2] Shahriar Asta and Ender Özcan. An apprenticeship learning hyper-heuristic for vehicle routing in hyflex. 12 2014.

[3] Aigerim Bogyrbayeva, Meraryslan Meraliyev, Taukekhan Mustakhov, and Bissenbay Dauletbayev. Learning to solve vehicle routing problems: A survey, 2022.

[4] Félix Chalumeau, Ilan Coulon, Quentin Cappart, and Louis-Martin Rousseau. Seapearl: A constraint programming solver guided by reinforcement learning, 2021.

[5] Przemysław Czuba and Dariusz Pierzchała. Machine learning methods for solving vehicle routing problems. 01 2021.

[6] Serap Ercan Cömert, Harun Yazgan, İREM SERTVURAN, and HANİFE ŞENGÜL. A new approach for solution of vehicle routing problem with hard time window: an application in a supermarket chain. *Sādhanā*, 42, 11 2017.

[7] Simone Foa, Corrado Coppola, Giorgio Grani, and Laura Palagi. Solving the vehicle routing problem with deep reinforcement learning, 2022.

[8] Waldy Joe and Hoong Chuin Lau. Deep reinforcement learning approach to solve dynamic vehicle routing problem with stochastic customers. In *International Conference on Automated Planning and Scheduling*, 2020.

[9] E. Queiroga, R. Sadykov, E. Uchoa, and T. Vidal. 10,000 optimal cvrp solutions for testing machine learning based heuristics. In *AAAI-22 Workshop on Machine Learning for Operations Research*, 2022.

[10] Pan Zhang. Evaluating accuracy of community detection using the relative normalized mutual information. *Journal of Statistical Mechanics: Theory and Experiment*, 2015(11):P11006, nov 2015.

[11] Xiang Zhang, Rui-Sheng Wang, Yong Wang, Jiguang Wang, Yu-Qing Qiu, Lin Wang, and Luonan Chen. Modularity optimization in community detection of complex networks. *EPL (Europhysics Letters)*, 09 2009.

[12] Ítalo Santana, Andrea Lodi, and Thibaut Vidal. Neural networks for local search and crossover in vehicle routing: A possible overkill?, 2022.