8-17-2023

Reinforcement Learning Approach to Stochastic Vehicle Routing Problem With Correlated Demands

Zangir Iklassov
Mohamed Bin Zayed University of Artificial Intelligence

Ikboljon Sobirov
Mohamed Bin Zayed University of Artificial Intelligence

Ruben Solozabal
Mohamed Bin Zayed University of Artificial Intelligence

Martin Takac
Mohamed Bin Zayed University of Artificial Intelligence

Recommended Citation

Z. Iklassov, I. Sobirov, R. Solozabal and M. Takáč, "Reinforcement Learning Approach to Stochastic Vehicle Routing Problem With Correlated Demands," in *IEEE Access,* vol. 11, pp. 87958-87969, 2023, doi: 10.1109/ACCESS.2023.3306076.

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2022.0122113

Reinforcement Learning Approach to Stochastic Vehicle Routing Problem with Correlated Demands

ZANGIR IKLASSOV, [1] IKBOLJON SOBIROV, [1] RUBEN SOLOZABAL, [1] AND MARTIN TAKÁČ [1]
[1]MBZUAI, Abu-Dhabi, UAE

ABSTRACT We present a novel end-to-end framework for solving the Vehicle Routing Problem with stochastic demands (VRPSD) using Reinforcement Learning (RL). Our formulation incorporates the correlation between stochastic demands through other observable stochastic variables, thereby offering an experimental demonstration of the theoretical premise that non-i.i.d. stochastic demands provide opportunities for improved routing solutions. Our approach bridges the gap in the application of RL to VRPSD and consists of a parameterized stochastic policy optimized using a policy gradient algorithm to generate a sequence of actions that form the solution. Our model outperforms previous state-of-the-art metaheuristics and demonstrates robustness to changes in the environment, such as the supply type, vehicle capacity, correlation, and noise levels of demand. Moreover, the model can be easily retrained for different VRPSD scenarios by observing the reward signals and following feasibility constraints, making it highly flexible and scalable. These findings highlight the potential of RL to enhance the transportation efficiency and mitigate its environmental impact in stochastic routing problems. Our implementation is available online.[a]

[a]https://github.com/Zangir/SVRP

INDEX TERMS Reinforcement Learning, Stopchastic Optimization, Vehicle Routing Problem

I. INTRODUCTION

Reinforcement Learning (RL) is a subdiscipline of machine learning that aims to teach algorithms to make decisions by maximizing a predefined reward signal through interactions with their environment. RL has demonstrated remarkable success across a range of applications, including gaming [33] and robotics [2]. One of the recent areas of research on RL is its application to combinatorial optimization problems. These problems have traditionally been solved using either fast but locally optimal heuristics or globally optimal but slow solvers (e.g., Google OR tools). The application of RL to combinatorial optimization offers a middle ground, providing a potential balance between computational efficiency and optimality, while also offering the ability to learn black-box heuristics without human intervention. Despite these benefits, the effective use of RL for combinatorial optimization requires careful design of the environment and appropriate training settings.

The Vehicle Routing Problem (VRP) is a well-known example of a combinatorial optimization problem. It is a generalization of the classic Traveling Salesman Problem (TSP) and involves finding the optimal route for a fleet of vehicles to deliver goods to customers with specified demands and locations. The objective was to minimize the total travel cost. The first RL model capable of solving the VRP was introduced in 2018 [25]. The authors employed a pointer-network-based architecture to tackle complex combinatorial challenges.

The Stochastic Vehicle Routing Problem (SVRP) is a challenging optimization problem with many potential applications in industry ([10], [13]). Unlike the deterministic VRP, some parameters in the SVRP are uncertain and can only be realized during the completion of the vehicle route. This formulation is more representative of real-world problems encountered in industry ([10], [13]). Although classical solutions have been proposed for the SVRP ([14], [21], [30]), the potential of Deep Reinforcement Learning to address this task has not yet been explored.

The SVRP with correlated stochastic demands is widely acknowledged as a highly challenging task within

the VRP research community ([31]). The solution to this problem has the potential to yield significant benefits in terms of cost and carbon emission reduction for companies involved in logistics and transportation. In this study, we aim to address the SVRP with correlated stochastic demands by developing a close-to-real-world formulation and solving it using an RL model.

Contributions. Our specific contributions to this field are summarized as follows:

- We propose a novel formulation for the Stochastic Vehicle Routing Problem with Correlated Demands (SVRPCD) that incorporates correlations among demands through observable stochastic variables. Our study focuses on optimizing a reinforcement learning model for this variant of VRP. Our proposed method exhibits an average performance improvement of 3.26% compared to the state-of-the-art model described in [14]. This advantage became more pronounced as the size of the problem increased.
- Our experimental results provide empirical evidence of the superiority of non-i.i.d. environments, a concept previously theorized in classical research. The state-of-the-art baseline model exhibited a 1% decrease in performance on instances with correlated demands, and this disparity increased with larger problem sizes. Conversely, our model demonstrated an improvement of 2% in instances with correlated demands, exhibiting consistent results across all problem sizes.
- Our model can learn the correlation between demands without human intervention, making it readily applicable to other SVRPCD settings. Through sensitivity experiments, we demonstrate the robustness of the model in response to environmental changes. This suggests the potential for significant impacts on transportation costs and environmental sustainability through additional research in this area. Further research could expand the scope of this study to incorporate additional sources of stochasticity, such as stochastic travel time and stochastic customers.

The subsequent sections of this paper are organised as follows. In Section II, we present an overview of the classical research avenues focused on addressing the SVRP, along with the state-of-the-art metaheuristic approach and previous endeavors to employ ML techniques for SVRP. Section III encompasses the classical problem formulation and provides definitions of the baseline models used in this study. Section IV elaborates on our problem formulation, encompassing the routing policy employed to train the RL agent, as well as the sensitive factors that influence the proposed model. Subsequently, in Section V, we provide a detailed description of the architecture employed. The experimental evaluation and assessment of robustness against sensitive factors are conducted in Section VI. Lastly, Section VII offers the concluding remarks for this study.

II. RELATED WORK

Several studies have applied RL techniques to address combinatorial optimization problems. For example, [7] were pioneers in utilizing RL to solve Traveling Salesman (TSP) and Knapsack problems. They reported better performance compared to heuristics, achieving results that were close to globally optimal solutions. [28] applied a Q-learning approach to solve a Beer Game problem. Their model can also be extended to address decentralized multi-agent combinatorial problems. Furthermore, [36] demonstrated that RL techniques can find optimal solutions for simple combinatorial problems such as Knapsack, Secretary, and AdWords problems. These studies highlight the potential of RL for combinatorial optimization.

Recent advancements in deterministic VRP have leveraged RL techniques. The first RL-based method for solving the VRP was introduced in [25]. [24] proposed an RL agent as a metaheuristic algorithm that guides a set of simple rules in the search for VRP solutions. This model demonstrated the lowest travel cost for VRP instances with 50-100 customers. However, the method requires domain expertise from a researcher to construct a set of feasible heuristics. To address this limitation, [38] developed an end-to-end RL-based solver, which achieved state-of-the-art results for large-scale VRP instances with 500, 1000, and 2000 customers. [1] presented a comprehensive analysis of the synergistic integration of classical VRP heuristics and contemporary ML methodologies. [18] introduced a novel hyperheuristic scheme called Bandit VNS. This scheme leverages the General Variable Neighborhood Search (VNS) metaheuristic to address deterministic VRP. Through experimental evaluations, the authors demonstrated that the Bandit VNS approach outperforms previous heuristics and RL methods in terms of solution quality. Specifically, for instances involving 32 to 120 customers and 5 to 25 vehicles, the proposed approach achieved a remarkable improvement of over 25%.

Despite extensive research on deterministic VRP utilizing RL algorithms, the application of Machine Learning (ML) algorithms to VRPSD remains limited. As discussed in [35], a significant body of literature exists only on classical algorithms for VRPSD, which are typically categorized into three main types:

1) Branch-and-Bound algorithms: Branch-and-Cut [12], Branch-and-Price [9], and Integer L-shaped [21].
2) Heuristics: LKH3 [15] and Clarke-Wright [30]
3) Metaheuristics: Tabu Search [22] and Ant-Colony Optimization [14]

Among Branch-and-Bound algorithms, Integer L-shaped [21] has demonstrated superior performance in solving VRPSD. The algorithm replaces the cost with variable θ, which is used to compute a lower bound through the application of the branch-and-bound algorithm. However, the computational time of the Integer L-shaped algorithm exhibits an exponential dependence on the number of vehicles, limiting its practical application in industry contexts. The literature on VRP has seen extensive research on the use of heuristics algorithms, with LKH3 showing superior performance for deterministic problems ([15]), but giving unstable results for SVRP. By contrast, the Clarke-Wright algorithm [30] is a simple heuristic that has shown stable performance for both deterministic and stochastic problems. Its longevity and consistency in the field have made the Clarke-Wright algorithm a frequently used reference point for comparative purposes in SVRP literature. The application of metaheuristic algorithms has been prevalent in industry for the solution of VRP with stochastic demands with Ant-Colony Optimization (ACO), demonstrating state-of-the-art performance [14].

A separate subfield of literature on VRPSD involves instances with correlated demands. To ensure the efficacy of classical methods for VRPSD instances with non-i.i.d. demands, it is crucial to consider demand correlation in the design of the algorithm. For instance, the pheromone update formula in ACO should be adjusted to reflect the correlation between demands ([31]). Similarly, Branch-and-Bound algorithms must be modified to address this correlation [8]. In general, classical algorithms have been shown to exhibit inferior performance on this type of problem because of the assumption of i.i.d. demands in their design.

Theoretically, instances of the Vehicle Routing Problem (VRP) with correlated stochastic demands have the potential to reduce travel costs compared to uncorrelated instances. This is because the model can exploit the correlation to improve demand estimation, and as a result, implement a more effective routing policy. This theoretical advantage has been previously established in [8], and its practical application was demonstrated in [31]. However, their approach required manual intervention to provide information about the demand probability density functions to the model, which limits its generalizability.

In recent years, a limited number of studies have applied ML algorithms to SVRP. In [32], a policy-based model was proposed to address the SVRP with stochastic demands, where the authors trained a simple linear model and achieved improved performance compared to heuristic methods. In [26], the authors utilized decision trees to solve the SVRP, whereas [27] employed an evolutionary algorithm based on a radial basis network. Furthermore, [16], [5] used deep reinforcement learning to address SVRP with stochastic customers. Additionally, [29] introduced a pioneering study utilizing RL to address Dynamic Uncertain VRP, where the demand is stochastic. However, in their context, the demand is uncorrelated, and its value is always known to the agent, albeit it can vary randomly during the routing procedure. In contrast, our study addresses a scenario with correlated demands that remain a priori unknown to the agents.

SVRP can find applications in various research domains, including the Internet of Vehicles (IOV), which integrates vehicles with the Internet of Things network and Smart City solutions. [3] presents a new IOV architecture that utilizes RL techniques based on vehicle2vec embeddings. The experimental results demonstrate that this architecture significantly reduces delivery delay by more than 20% when compared to state-of-the-art baselines. [4] proposes a T-Coin based IOV system that incentivizes vehicles to follow routes that alleviate traffic congestion. The method is evaluated on a simulated traffic scenario using a real map of Beijing city, and the results reveal a reduction in average travel time compared to the baselines. These research works illustrate the potential of employing VRP (especially SVRP) solutions in modern IOV and Smart City systems, offering promising avenues for addressing transportation challenges and improving overall efficiency.

III. BACKGROUND

The Vehicle Routing Problem (VRP) is an NP-hard combinatorial optimization problem. This involves optimizing the routes of vehicles to deliver goods from a depot node to customer nodes with known locations and demands. The objective of the VRP is to minimize the total travel cost of all vehicles used to meet customer demands. Over the years, VRP has been extensively studied and various methods and heuristics have been proposed to solve it [20]. The original VRP formulation was first introduced in [11].

The Stochastic Vehicle Routing Problem (SVRP) is a challenging optimization problem that is of increasing interest in the research community owing to its greater applicability to real-world industrial problems. Unlike its deterministic counterpart, the SVRP parameters are uncertain in advance and can only be realized during the completion of the route. This makes the SVRP a more representative model of real-world routing scenarios and has led to a growing body of research dedicated to solving it [10], [13]. According to [35], there are three main sources of stochasticity in the SVRP:

1) stochastic demands (VRPSD) - the demand of customers is uncertain,
2) stochastic customers (VRPSC) - the presence or absence of certain customers can be uncertain,
3) stochastic travel times (VRPSTT) - the costs associated with traversing routes are uncertain.

[23] demonstrated that deterministic formulation cannot provide an optimal solution to the SVRP. [21] presented the formulation of the VRPSD, providing a mathematical

representation that considers the inherent uncertainties in the problem.

A. CLASSICAL PROBLEM FORMULATION

Notations:
N : set of nodes (customers and depot)
C : set of customers
c_{ij} : cost of travel from node i to node j
ξ_i : stochastic demand variable of customer i
μ_{is} : s^{th} realization of the demand of customer i
K : total number of vehicles
Q : maximum capacity of each vehicle
x_{ij} : 1 if arc (i, j) is used in the route, 0 otherwise.

VRPSD is formulated on a complete, undirected graph $G = (N, X)$. The set of nodes $N = 0, 1, \ldots, n$ includes a depot node (denoted by node 0) and customer nodes. The depot node serves as the base for a fleet of K vehicles with an initial capacity of Q. Each customer node $i \in C = N \backslash 0$ has a stochastic demand variable ξ_i, with its s^{th} realization denoted as μ_{is}. Set of arcs $X = (i, j) : i, j \in N, i < j$ represents the connections between nodes, with the cost of traversing arc $(i, j) \in X$ in a route denoted by c_{ij}. The variable x_{ij} represents the action of an agent, where x_{ij} is 1 if arc (i, j) is used in the route, and 0 otherwise.

The solution to the problem comprises a set of routes assigned to each of the K vehicles, where each route is a sequence of nodes that start and end at the depot node (node 0). Upon visiting a customer node, the corresponding goods are supplied, thereby reducing the customer's demand to zero and decreasing the vehicle's capacity by the magnitude of the customer's demand.

The objective of the problem is to

$$\begin{aligned}
\text{minimize} \quad & \textstyle\sum_{i,j\in C} c_{ij}x_{ij} + \mathscr{R}(x), && (1)\\
\text{subject to} \quad & \textstyle\sum_{j=1}^{n} x_{0j} = 2|K|, && (2)\\
& \textstyle\sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2 \quad \forall k \in N, && (3)\\
& \textstyle\sum_{i,j\in C} x_{ij} \le |C| - [\sum_{i\in C} \mathbb{E}[\xi_i]/Q], && (4)\\
& 0 \le x_{ij} \le 1 \quad \forall i, j \in C, && (5)\\
& 0 \le x_{0j} \le 2 \quad \forall j \in N, && (6)\\
& x = (x_{ij}) \ \text{integer} \quad \forall i, j \in N. && (7)
\end{aligned}$$

The objective is to minimize the total travel cost (1), which is represented as the sum of the costs of each arc in the route. The constraints ensure that each vehicle's path begins and ends at depot (2), each customer is visited only once (3), the vehicle's maximum capacity is sufficient for the expected total demand (4), and decision variable x_{ij} is an integer for every arc (5), (6), (7). These constraints ensure that the solution is feasible and adheres to the underlying constraints of the problem.

Objective function (1) includes a term referred to as the recourse cost $\mathscr{R}(x)$. This term accounts for the cost associated with failure situations, where a vehicle may arrive at a customer node with insufficient capacity to fulfill demand. In such a scenario, the vehicle must return to the depot to refill its capacity and then return to the failed customer, incurring an additional cost that can be mathematically expressed as:

$$\begin{aligned}
\mathscr{R}(x) &= \textstyle\sum_{k=1}^{K} \mathscr{R}^k(x),\\
\mathscr{R}^k(x) &= 2\textstyle\sum_{j=2}^{t}\sum_{l=1}^{j-1} P(\sum_{s=2}^{j-1}\xi_s \le lQ < \sum_{s=2}^{j}\xi_s) c_{0j}.
\end{aligned}$$

The total recourse cost is equal to the sum of the recourse costs for each vehicle. The mathematical representation of the recourse cost of vehicle k is given as $\mathscr{R}^k(x)$. It calculates the likelihood of the l^{th} failure scenario for the j^{th} customer along the route.

B. BASELINES

The Clarke-Wright (CW) heuristic [30] is a well-known algorithm in the literature for VRPSD. This method is based on the concept of savings, which is a measure of the reduction in the total travel cost that results from merging two customer nodes into a single route. The saving value between customers i and j is calculated as

$$\text{saving}_{ij} = c_{0i} + c_{j0} - c_{ij},$$

where c_{0j} is the traveling cost between the depot and customer j, the same for customer i, and c_{ij} is the traveling cost between customer i and customer j. The savings values between all customer pairs are collected in a savings list and sorted in decreasing order. The route merging procedure begins by selecting the largest savings value from the top of the savings list. If the expected demand $\mathbb{E}[\xi_i] + \mathbb{E}[\xi_j]$ of combining customers i and j into the same route does not exceed vehicle capacity Q, then both customers are combined into the same route. The route merging procedure is repeated until no feasible mergings are possible in the savings list.

Tabu Search (TS) is a metaheuristic optimization algorithm that has been widely utilized for solving VRPSD. The algorithm begins with a randomly generated feasible solution and evaluates the total travel cost. Subsequently, over a defined number of iterations $k_{Tabu,max}$ a set of candidate solutions is generated by applying a set of predefined neighborhood heuristic operations to the current solution. Finally, the algorithm returns the best solution among all visited feasible solutions.

[22] shows that a specific set of four heuristics (specific neighborhood, 2-opt, swap-operation, reallocate-operation) demonstrates superior performance on VRPSD compared to other sets of heuristics that have been previously considered in the literature.

Ant Colony Optimization (ACO) is a well-established metaheuristic. It operates by simulating the behavior of ants in the search for food sources. The algorithm models the ants as solution agents that traverse the solution space by exploring and exploiting feasible routes. When a route is feasible, the corresponding solution agent lays a pheromone on the arcs that form the route. The concentration of pheromones in an arc is proportional

to the quality of the arc as a candidate solution, with higher pheromone concentrations leading to an increased likelihood of being followed by subsequent agents. Over time, the pheromone evaporates, simulating the natural behavior of ants. The search for new solutions continues as ants probabilistically select the arcs to follow.

The evaporation of pheromones helps avoid trapping the solution in the local optima. In a study by [14], the superiority of the ACO algorithm for VRPSD was demonstrated in comparison to other methods. The transition from the current location (i) to the next location (j) for an ant is governed by the following state transition formula,

$$\mathrm{j} = \mathrm{argmax}_{\mathrm{j}\in\varphi_\mathrm{i}} \left\{\tau_{\mathrm{ij}}^{\mathrm{a}} \eta_{\mathrm{ij}}^{\mathrm{b}}\right\}.$$

ACO algorithm is sensitive to its parameters: the pheromone trail τ_{ij} on the edge (i,j), the handcrafted heuristic η_{ij} that provides additional evaluation of the edge, the parameters a and b that control the relative importance of pheromone and heuristic, and the number of ants k utilized in the optimization process.

IV. METHOD

A. PROBLEM FORMULATION

Our formulation minimizes total travel cost over the set of demand realizations as follows

$$\text{minimize} \quad \tfrac{1}{|S|}\textstyle\sum_{\mathrm{s}\in\mathrm{S}}(\sum_{\mathrm{i},\mathrm{j}\in\mathrm{C}}\mathrm{c_{ij}x_{ij}} + 2\sum_{\mathrm{i}\in\mathrm{C}}\mathrm{r_i c_{0i}}),$$

s.t. $\forall \mathrm{i},\mathrm{j} \in \mathrm{N}$ the $\mathrm{r_i}, \mathrm{x_{ij}}$ are integers. Here $\mathrm{r_i}$ represents the recourse action of going to the depot, refilling, and returning to the $\mathrm{i^{th}}$ customer. The constraint implies that the recourse action must be an integer value. This formulation allows for the representation of complex VRPSD in a deterministic manner as a combination of scenarios, where each scenario s is characterized by the realization of the demand of each customer, μ_{is}. The set of scenarios S with demand realizations μ_{is} can be used for training purposes by any RL algorithm. This approach explicitly optimizes the recourse action, making the formulation suitable for solutions using both the classical and RL models.

Correlated demands. We incorporate new observable stochastic variables into the problem (weather variables), which are assumed to affect the stochastic demand of each customer, creating correlations between them. The agent could observe weather variables. Knowing their values, the agent can adjust its demand estimates and optimize its routing plan to minimize the objective. This formulation closely resembles industry practice, as companies can estimate the unknown demand for a given day by considering various parameters that influence it (e.g., weather). Each weather variable $\mathrm{w_j}$ is modeled to have a uniform distribution $\mathrm{w_j} \sim \mathrm{U}(-1;1)$ to affect the demand as follows

$$\xi_\mathrm{i} = \bar{\xi}_\mathrm{i} + \textstyle\sum_\mathrm{j}\sum_\mathrm{k}\alpha_{\mathrm{ijk}}\mathrm{w_j w_k} + \epsilon_\mathrm{i}.$$

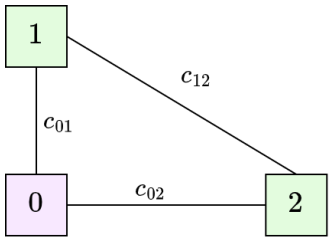


FIGURE 1: Correlated demands example. The example consists of a depot (node 0) and two customers (nodes 1 and 2). Let's assume that the travel cost from the depot to each customer is $\mathrm{c_{01}} = \mathrm{c_{02}} = 10$, and the travel cost between customers is $\mathrm{c_{12}} = 10\sqrt{2}$. Single vehicle with maximum capacity $\mathrm{Q} = 50$ is utilized. The demand of each customer can either be 10 or 50. In the case of uncorrelated demands, the optimal strategy is $0 \to 1 \to 0 \to 2 \to 0$, resulting in a travel cost of 40. However, if the demands are positively correlated with a 90% chance of having the same value $(10, 10)$ or $(50, 50)$, the vehicle can directly go to the second customer, if the demand of the first customer is 10, following the strategy $0 \to 1 \to 2 \to 0$. This results in an expected travel cost of 38. If the demands are 100% correlated, the expected travel cost is 37. Utilizing correlations among the demands can lead to improved performance of the agent.

We use $\mathrm{w_j}$ to model the stochastic demand ξ_i. The predicted value of the stochastic variable will be equal to a constant value $\bar{\xi}_\mathrm{i}$, augmented by the weighted cross effect of weather variables $\mathrm{w_j}, \mathrm{w_k}$, and perturbed by random noise ϵ_i correlated among customers. Weather variables are shared across customers, leading to correlated demand. Figure 1 illustrates a simple example of the impact of using correlated demands in the policy on agent performance.

B. LEARNING THE ROUTING POLICY

State. We define the RL setting as a VRPSD optimization problem with one vehicle and a given set of inputs at time t

$$\mathrm{I^t} \doteq \{(\mathrm{w}, \mathrm{p_i}, \mathrm{d_i^t}), \mathrm{i} \in \mathrm{N}\},$$

where each state is represented by a set of tuples. The tuple $(\mathrm{w}, \mathrm{p_i}, \mathrm{d_i^t})$ encodes information about the weather, location of customer i, and demand of customer i at time t. One can view w as a vector of features that can affect the demand of customers, $\mathrm{p_i}$ as a coordinate representation or one-hot encoding of the position of customer i, and $\mathrm{d_i^t}$ as the current demand estimate of customer i.

Within the context of Reinforcement Learning (RL), the action corresponds to the movement of a vehicle towards a specific node. Given the current state at time t, the agent selects the next node from a set N, and the vehicle proceeds to that node for the subsequent iteration at time t+1. Here, the term action is represented

by a pointer that chooses node $i \in N$ (which can be a customer or a depot) at each time step t using a policy π (i.e., $a^t \sim \pi(\cdot|I^t, h^t)$). The policy π takes into consideration the current state I^t and the memory state h^t, which encodes the information of all previous actions $a^1, ..., a^{t-1}$. By employing this policy π, a probability distribution over all nodes N is generated, enabling the agent to determine the node towards which the vehicle should move. When a specific action is executed, the vehicle incurs a corresponding cost denoted as C_t (in real-world scenarios, this cost can represent factors such as time, distance, or monetary expenses associated with this movement). In our RL framework, this cost is considered as the negative reward, implying that a lower cost corresponds to a higher reward for the agent.

Transition. We update the representation of the state after each action a^t is taken $I^{t+1} \sim f(\cdot|I^t, a^t)$, where the weather variables and customer locations are assumed to remain unchanged. Consequently, the formula for updating the customer demand d_i^t and vehicle capacity q^t provides a definition of the state transition function:

$$\begin{aligned} d_i^{t+1} &= \max\left\{0, d_i^t - q^t\right\}, \\ d_k^{t+1} &= d_k^t, \text{ for } k \neq i, \text{and } \; q^{t+1} = \max\left\{0, q^t - d_i^t\right\}. \end{aligned} \tag{8}$$

Objective. We quantify the cost at a given time step t after taking action a^t, which entails moving from node i to node j, as $C_t(I^t, a^t) = c_{ij}$. In the event of capacity exhaustion, the naive recourse strategy is employed, in which the vehicle returns directly to the depot to refill its capacity, incurring an additional cost $C_t(I^t, a^t) = c_{ij} + 2c_{0j}$. The policy is learned by utilizing the Reinforce algorithm [39] with the aim of minimizing the cumulative expected cost

$$\mathcal{J}^{\pi}(\theta) = \mathbb{E}[\textstyle\sum_t^T C_t^{\pi_\theta}(I^t, a^t)],$$

where θ represents the parameters of policy π and the cost of the policy is accumulated over the course of a finite horizon T, where T is defined as the point in time when all customer nodes have been visited. According to the Policy Gradient Theorem [34], the gradient of the objective function can be approximated as

$$\begin{aligned} \nabla_\theta \hat{\mathcal{J}}^{\pi}(\theta) \approx \tfrac{1}{B}\textstyle\sum_b^B\sum_{t=1}^T((TC(I^t, h^t) - b_\phi(I^t, h^t)) \\ \cdot \nabla_\theta \log \pi_\theta(a^t|I^t, h^t)), \end{aligned}$$

where b is an instance from a batch of size B; $b_\phi(s_t, x)$ is the baseline, the estimate of the value function, and TC, is the actual cumulative cost $TC(I^t, h^t) = \sum_t^T C_t(I^t, a^t)$. The baseline is computed using a critic network to speed up convergence. The critic is trained by updating the parameters ϕ to minimize

$$L(\phi) = \tfrac{1}{B}\textstyle\sum_b^B\sum_{t=1}^T ||b_\phi(I^t, h^t) - TC(I^t, h^t)||^2.$$

C. SENSITIVE FACTORS

The proposed method comprises several critical components that require careful consideration when assessing its robustness. First, the method's performance may be contingent on the choice of inference strategies used to determine the action a^t at each state. Second, the stochastic demand in the proposed method comprises three terms: a constant term, weather-related term, and noise. The absolute magnitude of each term can affect the performance of the method. Third, the maximum capacity of the vehicle may have a significant impact on the occurrence of failed situations. Fourthly, the stochastic demand can be estimated using various methods. Finally, in industry applications of the VRPSD, two additional factors can also be of critical importance: the agent may have access to the customer demand once he reaches its node, and the positions of the customers can be fixed.

1) Inference Strategies

Utilization of the greedy inference approach, where actions are selected based on the maximum probability $a^t = \arg\max[\pi(\cdot|I^t, h^t)]$, has been shown to produce suboptimal results [6], [19]. Therefore, this study evaluates two alternative inference strategies to address this issue.

The Sampling approach entails repeating the inference process several times by sampling from the policy distribution, with the aim of generating multiple routing solutions. The number of samples drawn is represented by the variable n_s. The Beam Search strategy [17], [37] involves exploring a set of n_b candidate actions at each time step of the inference process. Subsequently, each candidate solution is expanded with a new set of actions, from which the beam search selects only the n_b candidates with the highest likelihood.

2) Demand Terms Effect

To conduct a comprehensive analysis of the impact of each component of the stochastic demand formulation, we defined three term-effect variables:

$$A_i = \frac{\bar{\xi_i^2}}{T_i} \qquad B_i = \frac{\mathbb{E}\left[(\sum_j \sum_k \alpha_{ijk} w_j w_k)^2\right]}{T_i}, \qquad \Gamma_i = \frac{\mathbb{E}\left[\epsilon_i^2\right]}{T_i},$$

$$T_i = \bar{\xi_i^2} + \mathbb{E}[(\textstyle\sum_j \sum_k \alpha_{ijk} w_j w_k)^2] + \mathbb{E}\left[\epsilon_i^2\right],$$

where A_i, B_i, Γ_i represent the effects of the constant, weather-related, and noise terms on the observed demand realization. These values are expressed on a normalized scale $[0, 1]$ and are required to sum up to 1. Utilizing these variables can facilitate a more in-depth examination of the effect of the stochastic components of demand on the performance of the model.

3) Filling Rate

We define the filling rate $\Phi = \frac{Q}{\sum_i \mathbb{E}[\xi_i]}$, as the ratio of the maximum vehicle capacity to the sum of the expected demands of all customers. A higher value of Φ indicates a lower likelihood of failure demands, and thus reduces the need for recourse actions.

4) A Priori vs Reoptimization Routing

The term a priori route refers to a route solution that is generated prior to revealing the actual customer demands. This solution is based on assumed demand estimates $\hat{d}_i$, which model uses as an input demand, $d_i = \hat{d}_i, \forall i \in N$. The current vehicle capacity q^t is calculated with the given demand estimates. After a priori route is fully constructed the travel cost is calculated using the actual demand realizations.

The a priori approach is differentiated from the reoptimization approach, which involves updating the solution after the current demand realization of each customer has been revealed, $\hat{d}_i^t = \mu_{is}$. In the reoptimization approach, an initial demand estimate $\hat{d}_i$, is utilized for each customer i prior to the knowledge of its actual demand realization. Once the agent reaches customer i for the first time, the initial demand estimate is updated with the actual demand realization and utilized in subsequent optimization steps.

5) Demand Estimates

We introduce two methods for estimating the customer demand. The first method assumes that the unknown demand realization is equal to its constant component, $\hat{d}_i = \bar{\xi}_i, \forall i \in N$. The second approach for estimating demand involves utilizing information related to weather variables. This necessitates the existence of a weather history, which serves as a repository for tuples of the form (w^k, p_i, μ_{ik}), representing the k-th realization of weather and demand variables for customer i with position p_i. Given the actual weather value w, the Euclidean distance is calculated to identify a set of K nearest neighbors, N(w), in the weather history. The demand estimate is then derived from the normalized demands of the K nearest neighbors as follows:

$$\hat{d}_i = \left(\sum_{w_k \in N(w)}^{K} \frac{1}{||w_k - w||} \mu_{ik}\right) / \left(\sum_{w_k \in N(w)}^{K} \frac{1}{||w_k - w||}\right). \quad (9)$$

6) Fixed vs Flexible Customer Position

We consider two approaches to customer positioning. The first approach is a flexible position, in which customers may have different positions on the map for each new problem instance. In this case, the (z, l) coordinates on the map of each customer are used as static inputs to the model, $s_i = (z_i, l_i)$. Another approach is to maintain fixed customer positions. This scenario is common in real-world applications, where the locations of customers often remain constant. In this case, because the (z_i, l_i) coordinates are the same during training and inference, the customer location can be encoded using a one-hot vector $s_i = (0, 0, ..., 1, ..., 0)$.

7) Supply Type

Another factor that may affect the performance of the model is supply type. One possible approach is partial supply, where the demand of customer i can be met with any quantity of the current vehicle capacity q^t. In this scenario, the demand-state transition function remains the same (8). The second approach is full supply only, where the demand of customer i can be fulfilled only completely if there is a sufficient quantity of current vehicle capacity q^t. In this case, the state transition function of the demand of customer i upon the vehicle's visit to the customer's node will be defined as $d_i^{t+1} = 0$, if $q^t \geq d_i^t$ and $d_i^{t+1} = d_i^t$ otherwise.

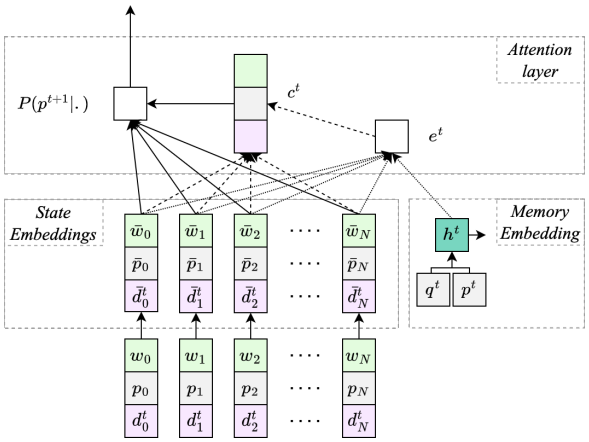


FIGURE 2: Network architecture. The bottom part depicts the input of the model that consists of three components: weather information w, the location p_i, and the dynamic demand d_i^t of each customer i. The model generates State Embeddings for each customer and encodes the vehicle's current position (p^t) and capacity (q^t) as the Memory Embedding (h^t). The embeddings are then combined through an Attention Layer to obtain probabilities over the nodes, representing the likelihood of each node being the next position of the vehicle.

V. ARCHITECTURE DETAILS

Embeddings. The architecture of the model is illustrated in Figure 2. It consists of three main components: finding state embeddings given the input data, encoding the sequence of visited nodes into a memory embedding, and decoding concatenated embeddings into action probabilities using the attention layer. The input to the model is a set of tuples (w, p_i, d_i^t) for each node i. The state embeddings are obtained by applying a convolution layer, resulting in a D-dimensional representation $\rho_i^t = (\bar{w}, \bar{p}_i, \bar{d}_i^t)$.

Encoding. The encoding of the sequence of visited nodes is accomplished using LSTM cell. This cell produces a memory embedding $h^t \in \mathbb{R}^D$, given the vehicle's current position p^t and capacity q^t. To combine the information from the different embeddings, we utilize an attention mechanism that computes an attention score $e^t = e^t(\rho_i^t, h^t)$ and a context vector $c^t = \sum_{i=1}^{M} e_i^t \rho_i^t$.

Decoding. The conditional probabilities are computed by combining context vector c^t and state embeddings as

$$u_i^t = W_a^T \tanh\left(W_b[\rho_i^t; c^t]\right), P(p^{t+1}|\cdot) = \text{softmax}(u_i^t),$$

where W_a and W_b are the trainable parameters. The softmax function is applied to the values u_i^t to obtain a vector of probabilities $P(p^{t+1}|\cdot)$ representing the likelihood of each node being the next in the route. These probabilities can then be utilized in the inference strategy to produce the routing solution.

VI. EXPERIMENTATION

A. DATA GENERATION

Positions. In this study, we evaluated the performance of model for five different problem sizes, with 10, 20, 50, and 100 customers. The problem instances were generated by randomly selecting the coordinates (z_i, l_i) of each customer within the unit square $[0, 1] \times [0, 1]$. The depot was located at the center of this square with coordinates $(z_0, l_0) = (0.5, 0.5)$.

Demand. Three weather variables are considered in this study: temperature, humidity, and wind speed. These variables were generated using a uniform distribution in the interval $[-1, 1]$. The constant part of the demand $\bar{\xi}_i$ is generated using a uniform distribution over the interval $[1, 10]$, while the noise part ϵ follows a multivariate normal distribution $\mathcal{N}(0, \Sigma)$, where variance $\sigma^2_{\epsilon_i}$ is chosen based on a given noise term effect value Γ_i and a randomly created non-zero covariance matrix Σ. By default, the values of A_i, B_i, and Γ_i are set to 0.6, 0.2, and 0.2 respectively. Then, the values α_{ijk} are randomly generated to satisfy the given B_i, $\bar{\xi}_i$, and w_j, w_k values. Finally, the capacity Q was determined based on the filling rate Φ.

Datasets. To obtain the weather dataset, a set of 10,000 realizations of (w, p_i, μ_{is}) was generated using the above (positions and demand) generation process. During the model training phase, random values (w, p_i, μ_{is}) is created for every training iteration. In addition, a validation dataset consisting of 1,000 instances was created for each problem size and saved for ACO parameter search. A similar process was used to create a test dataset for model comparison.

B. MASKING

A masking procedure was implemented to generate feasible solutions. This procedure involves setting the log-probabilities of infeasible solutions to $-\infty$. The masking we used included the following:

- preventing already visited nodes to be visited again
- preventing all customer nodes to be visited if the vehicle's estimated capacity is 0
- preventing customers whose estimated demand exceed the estimated vehicle capacity

This masking procedure ensures that when the vehicle visits a customer, it must satisfy all customer demands.

C. IMPLEMENTATION DETAILS

Parameters. To obtain the state embedding, we employed a 1-dimensional convolutional layer with D filters and $|\rho_i^t|$ channels. To obtain the memory embedding, we utilized single-layer LSTM with a state size D. The outputs of the layer computing $P(p^{t+1}|\cdot)$ are followed by a critic network, consisting of two fully-connected layers with D and one neurons respectively. The activation function utilized in all the layers is the Rectified Linear Unit (ReLU). The default dimensionality of the embeddings was set to $D = 128$.

Training. The variables in the neural network architecture were initialized using the Xavier initialization method. During training, the Adam optimizer was used with a learning rate of 10^{-4} and a batch size of $B = 128$. To reduce overfitting, a dropout with a probability of 0.1 was employed during training. The model was trained on a GPU system consisting of an NVIDIA A100 SXM 40GB GPU and 2x AMD EPYC 7742 CPUs (8 cores) with 256GB RAM for 10,000 iterations on each problem size.

D. RESULTS

a: Baselines.

For the CW heuristic, implementation proposed in [30] was employed. For the Tabu search, we used [22] algorithm, with a set of four heuristics mentioned in the paper and a maximum of 2000 iterations. The ACO algorithm implementation [14] is sensitive to its parameter values, and thus, a random grid search was performed over discrete parameter space, with the pheromone importance (a) in the range [1, 10], heuristic importance (b) in the range [1, 10], and the number of ants (c) in the range [5, 30]. The performances of 15 randomly generated parameter tuples were evaluated using a validation dataset. Table 1 presents the results for five parameter tuples, with the best result achieved by a,b, and c equal to 3,10, and 12 respectively. For the heuristic evaluation, the reverse distance measure, $\eta_{ij} = \frac{1}{d_{ij}}$, was employed.

Table 2 presents a comparison of the average travel cost results on the test data for the baselines and RL models. The baselines consist of Tabu search, CW, and the best-performing model of ACO, as listed in Table 1. According to classical research findings ([14]), the ACO model consistently exhibits the lowest travel cost across all problem sizes among the baselines.

For the RL model, an a priori routing approach was employed with the demand values estimated using the k-NN algorithm ($k = 5$). This configuration provides a fair comparison with the baselines, as their routing solutions are determined in an a priori manner and they use the same demand estimates. During inference, a beam search strategy with a beam width ($n_b = 3$) was utilized. The RL model exhibits improved performance compared to ACO, resulting in a decrease in the travel cost by 1.26%,

TABLE 1: Results of ACO algorithm with varying parameter values on SVRP validation data. The rows represent values for the parameters controlling the pheromone importance (a), heuristic importance (b), and number of ants (k). The columns indicate the number of customers. Results are reported as the average travel cost over the validation data, with lower values indicating better performance.

ACO parameters	10	20	50	100
3 / 4 / 20	3.24	7.21	14.11	31.26
6 / 8 / 24	3.21	7.05	14.02	31.13
8 / 2 / 5	3.23	7.11	14.19	31.25
4 / 9 / 15	3.18	6.79	14.03	31.14
3 / 10/ 12	3.17	6.54	13.92	30.89

TABLE 2: Results of Baseline and the RL models on SVRP test data. The rows depict three VRPSD heuristic/metaheuristics and RL model that employs a beam search inference strategy.

Baselines/Instances	10	20	50	100
Clarke-Wright	3.31	7.31	14.54	32.59
Tabu Search	3.25	7.25	14.23	31.64
Ant-Colony Opimization	3.17	6.54	13.92	30.89
RL-beam search	3.13	6.39	13.33	29.27

2.29%, 4.24%, and 5.24% for 10, 20, 50, and 100 customers respectively. On an average, the observed improvement was 3.26%. It is worth mentioning that a larger size of the problem constitutes a larger % gap between the RL and ACO models.

b: Sensitive Factors.

The following experiments were conducted to assess the sensitivity of the proposed model to various factors. The default model utilized RL with a beam-search inference strategy ($n_b = 3$), k-NN demand estimates, fixed customer positions, partial supply, demand term proportions set to $A, B, \Gamma = 0.6, 0.2, 0.2$, and filling rate $\Phi = 0.5$. We started experiments with a single-vehicle scenario to evaluate the model's performance and its dependence on sensitive factors. In each experiment, a single factor was changed and evaluated across the four problem sizes, and the results for both the a priori and reoptimization approaches are reported. Although the reoptimization approach consistently produced superior results, the a priori approach is still relevant in industrial applications and may be effective in certain real-world scenarios; thus, both approaches are important to consider.

The performance of greedy, sampling, and beam-search inference strategies were evaluated. The results (Table 3) indicate that beam-search exhibits superior performance across all four problem sizes using both a priori and reoptimization approaches. Sampling strategy with $n_s = 16$ was found to be superior to greedy search.

Experiments evaluating the demand estimates were performed, and the results indicate superior performance of the k-NN estimate, as shown in Table 4. The results of the experiments investigating the impact of customer positions on performance are presented in Table 5. The results indicate the superior performance of fixed customer positions in the test data.

Four sets of values were utilized in the experiments on different demand term effects (Table 6). The results indicate that constant and weather terms have a positive impact on performance, whereas noise has a negative effect.

The Figure 3 illustrates the percentage gap in travel costs between the correlated and uncorrelated demands. For the purpose of these experiments, correlated demands were generated using the values $A, B, \Gamma = 0.8, 0.2, 0.0$, while uncorrelated demands were generated using $A, B, \Gamma = 0.8, 0.0, 0.2$. The results reveal that the RL model with the reoptimization approach yields a 2.2% improvement in performance when dealing with correlated demands, and this difference is consistent across all problem sizes. The RL with an a priori approach

TABLE 3: Results of different Inference strategies on SVRP test data. The rows depict three different inference strategies. The columns indicate number of customers. Results are reported for a priori (Apr.) and reoptimization (Reopt.) settings.

	10		20		50		100	
Inference strategies	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.
RL-greedy	3.60	3.49	7.35	7.13	15.33	14.87	33.66	32.65
RL-sampling	3.32	3.22	6.78	6.57	14.13	13.71	31.03	30.10
RL-beam-search	3.13	3.04	6.39	6.20	13.33	12.93	29.27	28.39

TABLE 4: Results of two types of demand estimates on SVRP test data. The rows depict demand estimate types.

	10		20		50		100	
Demand Estimate	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.
Constant demand estimate	3.38	3.28	6.90	6.70	14.40	13.96	31.61	30.67
kNN demand estimate	3.13	3.04	6.39	6.20	13.33	12.93	29.27	28.39

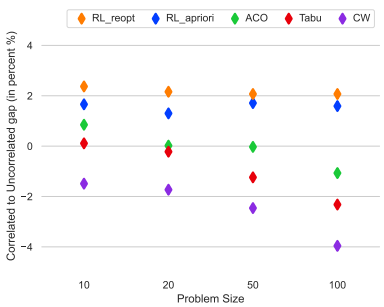


FIGURE 3: Avg. travel cost gap (in percent) of setting with correlated demands ($A/B/\Gamma = 0.8/0.2/0.0$) to setting with uncorrelated demands ($A/B/\Gamma = 0.8/0.0/0.2$) for Baseline and RL models with higher values indicating better performance.

TABLE 5: Results of two types of Customer positions on SVRP test data. The rows depict customer position types.

	10		20		50		100	
Customers Position	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.
Flexible customer positions	3.51	3.40	7.16	6.95	14.93	14.48	32.78	31.80
Fixed customer positions	3.13	3.04	6.39	6.20	13.33	12.93	29.27	28.39

TABLE 6: Results of different proportions of demand terms on SVRP test data. The rows depict proportional effect of demand terms.

	10		20		50		100	
A/B/Γ	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.
0.8 / 0.0 / 0.2	3.02	2.95	6.15	6.02	12.88	12.54	28.26	27.54
0.8 / 0.2 / 0.0	2.97	2.88	6.07	5.89	12.66	12.28	27.81	26.97
0.6 / 0.2 / 0.2	3.13	3.04	6.39	6.20	13.33	12.93	29.27	28.39
0.4 / 0.3 / 0.3	3.38	3.28	6.90	6.70	14.40	13.96	31.61	30.67

provides a stable, albeit lower, improvement on average. On the other hand, the baseline models display a negative gap, indicating that they do not capture the correlations between demands, and this negative gap becomes more pronounced with larger problem sizes.

The experiments on signal ratio employed the values of Φ equal to $0.1, 0.5,$ and 0.9. The results presented in Table 7, indicate that Φ has a positive impact on performance of the model, where higher Φ values result in increased vehicle capacity and fewer recourse actions. Importantly, even with a relatively small value of $\Phi = 0.1$, the model is still able to produce comparably good performance, despite the need for an average of ten recourse actions.

Experiments evaluating the influence of supply type on performance were conducted, and the results are presented in Table 8. The data indicate that partial supply conditions result in the superior performance on the test data.

E. MULTI-VEHICLE SCENARIO

To scale our approach to a multi-vehicle scenario, we augmented the input layer of the actor network with q_k^t and p_k^t for all vehicles $k \in K$ and predicted p_k^{t+1} for each vehicle. This enabled us to find the next position of each vehicle based on its current position and capacity with minimal modifications to the architecture.

We tested the model on varying numbers of vehicles

TABLE 7: Results of different Filling rates on SVRP test data. The rows depict values of the filling rate.

	10		20		50		100	
Filling Rate	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.
0.1	3.51	3.40	7.16	6.95	14.93	14.48	32.78	31.80
0.5	3.13	3.04	6.39	6.20	13.33	12.93	29.27	28.39
0.9	2.91	2.82	5.95	5.77	12.40	12.02	27.22	26.41

TABLE 8: Results of two Supply types on SVRP test data. The rows depict supply types.

	10		20		50		100	
Supply Type	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.	Apr.	Reopt.
Full supply only	3.47	3.37	7.10	6.88	14.80	14.35	32.49	31.52
Partial supply	3.13	3.04	6.39	6.20	13.33	12.93	29.27	28.39

TABLE 9: Results of a varying number of vehicles on SVRP test data. The rows depict number of vehicles.

Number of vehicles	10	20	50	100
1	3.04	6.20	12.93	28.39
2	2.98	6.13	12.74	28.01
3	2.95	6.06	12.59	27.73
5	2.95	6.01	12.42	27.64

TABLE 10: Average solution time (in seconds) using different baselines over a test set of size 1000.

Baseline	10	20	50	100
Clarke-Wright	0.003	0.013	0.054	0.171
Tabu Search	0.831	2.723	8.123	56.35
Ant-Colony Optimization	1.028	4.095	27.39	120.3
RL-greedy	0.052	0.108	0.159	0.328
RL-sampling	0.059	0.129	0.204	0.385
RL-beam search	0.064	0.141	0.217	0.406

using the RL model with a reoptimization setting. The results (Table 9) suggest that increasing the number of vehicles can lead to a decrease in total travel cost, but its marginal effect diminishes as the number of vehicles increases. This demonstrates the ability of the model to obtain better routing strategies when having more vehicles to operate with.

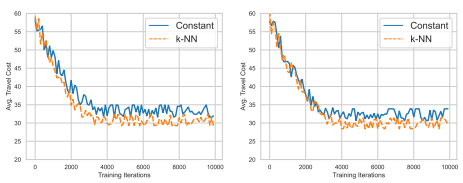
F. TIME COMPLEXITY

Table 10 displays a comparison of the average run-times of the algorithms considered. The RL models adopted an a priori routing approach, with the k-NN algorithm ($k = 5$) used for demand estimation. This setup ensures a fair comparison with the baselines (CW, Tabu, and ACO), as their routing solutions are determined in an a priori manner, and they use the same demand estimates.

Among the considered algorithms, the CW heuristic exhibited the shortest average time required to obtain a solution for a given instance. Although RL requires more time, all the solutions were found in less than one second. It is noteworthy that both RL and CW exhibit linear time complexity, whereas Tabu and ACO display quadratic complexity.

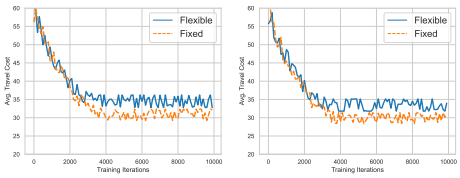
G. TRAINING CURVES

The training curves of different demand estimates can be observed in Figure 4, where both approaches initially exhibit similar performance, but after 2000 iterations,



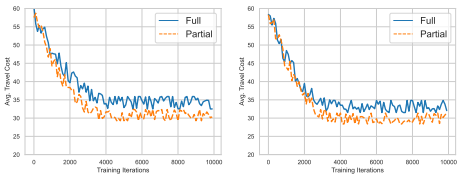
(a) A priori (b) Reoptimization

FIGURE 4: Average travel cost observed during training using (a) a priori and (b) reoptimization approaches of constant and k-NN demand estimates. The k-NN estimate outperforms the constant in both settings.



(a) A priori (b) Reoptimization

FIGURE 5: Average travel cost observed during training using (a) a priori and (b) reoptimization approaches for Flexible and Fixed customer positions. Fixed positions outperform Flexible positions in both settings.



(a) A priori (b) Reoptimization

FIGURE 6: Average travel cost observed during training using (a) a priori and (b) reoptimization approaches of Full and Partial supply types. Partial supply outperforms Full supply in both settings.

the k-NN estimate consistently demonstrates improved results. Figure 5 shows the training curves of the different customer positioning approaches, which depict the consistently improved performance of fixed customer positions during the training process. Figure 6 shows the training curves of different supply types.

VII. DISCUSSION AND CONCLUSION

In this study, we propose a novel formulation for the Stochastic Vehicle Routing Problem with Correlated Demands (SVRPCD) that incorporates the impact of weather variables on customer demands. This formulation is relevant for practical applications because it enables companies to consider observed variables, such as weather, in the planning of vehicle routes. Our proposed approach utilizes Reinforcement Learning (RL) to learn the correlation between demands, making it highly applicable to various SVRPCD scenarios.

We conducted experiments to evaluate the performance of the RL model compared with the state-of-the-art model in [14]. Our results demonstrate that the RL model outperforms the prior approach and confirm the superiority of non-i.i.d. environments for VRPSD problems. Furthermore, the model can learn the correlation between demands without human intervention, which enhances its practicality and versatility. Through sensitivity experiments, we demonstrate the robustness of the model in response to environmental changes.

The findings of this study suggest potential for significant reductions in transportation costs and improvements in environmental sustainability. Future research could extend the scope of this study to include additional sources of stochasticity, such as stochastic travel time.

REFERENCES

[1] Luca Accorsi, Andrea Lodi, and Daniele Vigo. Guidelines for the computational testing of machine learning approaches to vehicle routing problems. Oper. Res. Lett., 50:229–234, 2021.

[2] Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub W. Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Joshua Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. The International Journal of Robotics Research, 39:20 – 3, 2020.

[3] Nyothiri Aung, Sahraoui Dhelim, Liming Chen, Abderrahmane Lakas, Wenyin Zhang, Huansheng Ning, Souleyman Chaib, and Mohand Tahar Kechadi. Vesonet: Traffic-aware content caching for vehicular social networks using deep reinforcement learning. IEEE Transactions on Intelligent Transportation Systems, pages 1–12, 2023.

[4] Nyothiri Aung, Weidong Zhang, Sahraoui Dhelim, and Yibo Ai. T-coin: Dynamic traffic congestion pricing system for the internet of vehicles in smart cities. Information, 11(3), 2020.

[5] Rafael Basso, Balázs Kulcsár, Iván D. Sánchez-Díaz, and Xiaobo Qu. Dynamic stochastic electric vehicle routing with safe reinforcement learning. Transportation Research Part E: Logistics and Transportation Review, 2022.

[6] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. arXiv preprint arXiv:1611.09940, 2016.

[7] Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. ArXiv, abs/1611.09940, 2017.

[8] Federica Bomboi, Christoph Buchheim, and Jonas Pruente. On the stochastic vehicle routing problem with time windows, correlated travel times, and time dependency. 4OR, pages 1–23, 2021.

[9] Christian H. Christiansen and Jens Lysgaard. A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. Oper. Res. Lett., 35:773–781, 2007.

[10] Jean-François Cordeau, Gilbert Laporte, Martin WP Savelsbergh, and Daniele Vigo. Vehicle routing. Handbooks in operations research and management science, 14:367–428, 2007.

[11] George B. Dantzig and John Hubert Ramser. The truck dispatching problem. Management Science, 6:80–91, 1959.

[12] Charles Gauvin, Guy Desaulniers, and Michel Gendreau. A branch-cut-and-price algorithm for the vehicle routing problem with stochastic demands. Comput. Oper. Res., 50:141–153, 2014.

[13] Michel Gendreau, Gilbert Laporte, and René Séguin. Stochastic vehicle routing. European Journal of Operational Research, 88(1):3–12, 1996.

[14] Rajeev Kumar Goel, Raman Maini, and Sandhya Bansal. Vehicle routing problem with time windows having stochastic customers demands and stochastic service times: Modelling and solution. J. Comput. Sci., 34:1–10, 2019.

[15] Keld Helsgaun. An extension of the lin-kernighan-helsgaun tsp solver for constrained traveling salesman and vehicle routing problems: Technical report. 2017.

[16] Waldy Joe and Hoong Chuin Lau. Deep reinforcement learning approach to solve dynamic vehicle routing problem with stochastic customers. In ICAPS, 2020.

[17] Chaitanya K Joshi, Thomas Laurent, and Xavier Bresson. An efficient graph convolutional network technique for the travelling salesman problem. arXiv preprint arXiv:1906.01227, 2019.

[18] Panagiotis Kalatzantonakis, Angelo Sifaleras, and Nikolaos Samaras. A reinforcement learning-variable neighborhood search method for the capacitated vehicle routing problem. Expert Systems with Applications, 213:118812, 2023.

[19] Wouter Kool, Herke Van Hoof, and Max Welling. Attention, learn to solve routing problems! arXiv preprint arXiv:1803.08475, 2018.
[20] Gilbert Laporte. Fifty years of vehicle routing. Transp. Sci., 43:408–416, 2009.
[21] Gilbert Laporte and François V. Louveaux. The integer l-shaped method for stochastic integer programs with complete recourse. Oper. Res. Lett., 13:133–142, 1993.
[22] Guoming Li and Junhua Li. An improved tabu search algorithm for the stochastic vehicle routing problem with soft time windows. IEEE Access, 8:158115–158124, 2020.
[23] François V. Louveaux. An introduction to stochastic transportation models. 1998.
[24] Hao Lu, Xingwen Zhang, and Shuang Yang. A learning-based iterative method for solving vehicle routing problems. In ICLR, 2020.
[25] Mohammadreza Nazari, Afshin Oroojlooy, Lawrence V Snyder, and Martin Takáč. Reinforcement learning for solving the vehicle routing problem. In Conference on Neural Information Processing Systems, NeurIPS 2018, 2018.
[26] Yunyun Niu, Detian Kong, Rong Wen, Zhiguang Cao, and Jian hua Xiao. An improved learnable evolution model for solving multi-objective vehicle routing problem with stochastic demand. Knowl. Based Syst., 230:107378, 2021.
[27] Yunyun Niu, Jie Shao, Jian hua Xiao, Wen Song, and Zhiguang Cao. Multi-objective evolutionary algorithm based on rbf network for solving the stochastic vehicle routing problem. Inf. Sci., 609:387–410, 2022.
[28] Afshin Oroojlooyjadid, M. Nazari, Lawrence V. Snyder, and Martin Takác. A deep q-network for the beer game: Deep reinforcement learning for inventory optimization. Manuf. Serv. Oper. Manag., 24:285–304, 2022.
[29] Weixuan Pan and Shi Qiang Liu. Deep reinforcement learning for the dynamic and uncertain vehicle routing problem. Applied Intelligence, 53:405–422, 2022.
[30] Tantikorn Pichpibul and Ruengsak Kawtummachai. A heuristic approach based on clarke-wright algorithm for open vehicle routing problem. The Scientific World Journal, 2013, 2013.
[31] Mojtaba Rajabi-Bahaabadi, Afshin Shariat, Mohsen Babaei, and Daniele Vigo. Reliable vehicle routing problem in stochastic networks with correlated travel times. Operational Research, 03 2021.
[32] Nicola Secomandi. Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. Comput. Oper. Res., 27:1201–1225, 2000.
[33] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. nature, 550(7676):354–359, 2017.
[34] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. Advances in neural information processing systems, 12, 1999.
[35] Paolo Toth and Daniele Vigo. Vehicle routing: Problems, methods, and applications, second edition. 2014.
[36] Learns Old Tricks. A new dog learns old tricks : Rl finds classic optimization algorithms. 2018.
[37] Libing Wang, Xin Hu, Yin Wang, Sujie Xu, Shijun Ma, Kexin Yang, Zhijun Liu, and Weidong Wang. Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning. Computer Networks, 190:107969, 2021.
[38] Q. Wang. Alpha-t: Learning to traverse over graphs with an alphazero-inspired self-play framework. 2021.
[39] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3):229–256, 1992.

ZANGIR IKLASSOV received an M.Sc. degree in economics from Nazarbayev University, Astana, Kazakhstan. He is currently pursuing a Ph.D. degree in Machine Learning from Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE. His research interests include deep reinforcement learning, artificial intelligence, and its applications. He has over three years of experience in Machine Learning Engineering.

IKBOLJON SOBIROV received his M.Sc. degree in Computer Vision from Mohamed bin Zayed University of Artificial Intelligence (MBZUAI) Abu Dhabi, UAE, where he is currently working as a research assistant. He also holds a B.Sc. degree in Business Information Systems from Westminster International University in Tashkent, Tashkent, Uzbekistan.

RUBEN SOLOZABAL received his Ph.D. degree in Information Technologies from the University of the Basque Country in 2021. He is currently a post-doctoral researcher in the Machine Learning Department at the Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE. His current research interests include Reinforcement Learning and Machine Learning in bioinformatics.

MARTIN TAKÁČ is currently an Associate Professor at the Mohamed bin Zayed University of Artificial Intelligence (MBZUAI), UAE. Before joining MBZUAI, he was an Associate Professor in the Department of Industrial and Systems Engineering at Lehigh University, where he has been employed since 2014. He received his B.S. (2008) and M.S. (2010) degrees in Mathematics from Comenius University, Slovakia, and Ph.D. (2014) degree in Mathematics from the University of Edinburgh, United Kingdom. His current research interests include designing and analyzing algorithms for machine learning, AI for science, understanding protein-DNA interactions, and using ML for energy. Martin received funding from various U.S. National Science Foundation programs (including through a TRIPODS Institute grant awarded to him and his collaborators at Lehigh, Northwestern, and Boston University) and recently was awarded a grant with the Weizmann Institute of Science. He served as an Associate Editor for Mathematical Programming Computation, Journal of Optimization Theory and Applications, Optimization Methods and Software, and as an area chair for ICLR and AISTATS. Martin currently serves as an area chair at ICML and NeurIPS.