Solution Approach to Resolve Capacitated Vehicle Routing Problem Using Deep Reinforcement Learning

A solution methodology that uncovers Deep Reinforcement Learning for solving Capacitated Vehicle Routing Problems.

AFFINE
COMMAND THE NEW

www.affine.ai

Contents

1. Introduction

Vehicle Routing Problem (VRP) -VRP can be defined as a problem for finding the optimum routes for a given set of vehicles to fulfil delivery and collection for a specified set of customers based on some pre-defined demand. Finding the optimal routes may involve business constraints like serving each customer only once. Our solution focuses on mapping the optimal routes for a single vehicle; hence the problem reduces to a simple Travelling Salesman Problem (Rani & Kumar, 2014). Optimal routes serve the purpose of minimizing the overall transportation cost, minimizing the number of vehicles, minimum distance travelled, minimum travel time, or other objectives. Some of the most important applications of VRP are Supply Chain Management, Mail delivery, Bus and railway route optimization, vehicle optimization, etc.

One of the variants of VRP is Capacitated VRP (CVRP). CVRP states that ***m*** capacitated vehicles initially at depot locations are required to fulfil discrete demands ***n*** customer nodes. The objective is to design the set of least distance paths with known customer demands. Refer Fig 3 for better understanding.

CVRP is one of the most popular problems in Network Optimization and is classified as an NP-hard problem. It means the size of the problem that can be solved optimally using mathematical programming or combinatorial optimization may be limited. Hence, most of the commercial solutions tend to use a meta-heuristic approach to solve real-world VRPs.

In this whitepaper, we have discussed one of the effective ways to solve the CVRP problem using Deep Reinforcement Learning. Below are the highlights of the process:

- Benchmarked the results of DeepRL against a popular algorithm called Genetic Algorithm
- Compared the performance using 2 KPIs – Travelling cost (distance covered) and Computational time
- During the comparison of KPIs performance, we have observed a 5X-20X reduction in cost and a 100X–1000X reduction in computational time
- The overall solution adheres to an adaptive learning framework where the system itself identifies the optimal solution without explicitly programming

2. Algorithms for Solving CVRP

CVRP is an NP-hard optimization problem, and various exact and approximation algorithms have been used to solve it. The algorithms are designed based on linear programming techniques, like Branch and Bound, Branch and Cut; the approximate algorithms are heuristic based viz. Genetic algorithms, Evolutionary algorithms, Tabu search, Dynamic Programming, and Neural networks, etc.

For *n* customer nodes and one depot nodes, there are n! numbers of paths that exist. For large CVRP instances, using exact methods for solving CVRP is computationally expensive and time-consuming. Due to this reason, approximate algorithms that employ heuristic search are often employed to solve CVRP problems. The choice of using exact vs. approximate algorithms is determined by the trade-off between optimality and computational cost.

Following are the recent important algorithmic developments for solving the CVRP problem in chronological order:

i) Pointer Networks (Vinyals et al., 2015): Pointer Networks (PN) is a kind of Recurrent Neural Networks that can be trained and produce outputs of variable lengths, which is vital in the case of VRP as the route lengths vary. The PN is used in a supervised way to find near-optimal tours to Travelling Salesman Problem from the given ground truth optimal solutions.

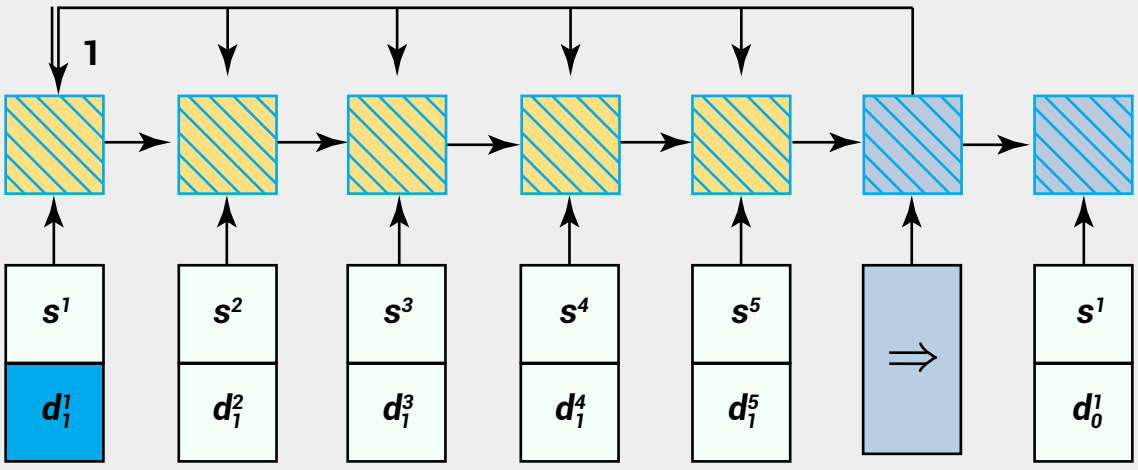

Fig.1: Pointer Networks (Figure taken from Nazari et al. 2018)

The drawback of this architecture is its sensitivity to small changes in the sequences. Due to which it gives inconsistent results. For instance - in Fig 1, if the demand d_1 of node S^1 is partially / completely fulfilled, the change should be incorporated into the entire network. For this, the entire Pointer network is required to be updated to compute the probabilities in the next decision point (with a new d_1 value). Another drawback is the dependence on supervision, which prohibits PN from finding better solutions than the ones provided during training.

ii) Actor-Critic Networks (Bello et al., 2016): The actor-critic model proposed by Bello et al. is the first work that lays the foundation for using reinforcement learning and neural networks to model CVRP.

- It is a pointer network which is trained without supervised solutions
- The drawback of this architecture is that it assumes the systems to be static over time. However, CVRP should include the dynamic nature of the problem, i.e. suppose the demand of the customer node is fulfilled, its demand should be updated in the network. This is the generic problem in Pointer Networks, and it is similar to the scenario discussed in the previous section

iii) Reinforcement Learning for CVRP (Nazari et al., 2018):

- This method is an extension of the Actor-Critic method (Bello et al., 2017), and it solves dynamic CVRP, which considers customer node's demand changes over time
- Long Short-Term Memory (LSTM) encoder in the pointer networks is replaced by element-wise projections. These are *invariant to the input sequence ordering* and will discard off unnecessary sequential information. Changing the order of any two inputs will not affect the network
- This framework shown in Fig 2 has an RNN decoder with an attention mechanism that is capable of handling both static and dynamic elements of the system and does the combinatorial optimization using both components
- The reward is calculated based on the generated outputs, thus if we can verify the feasibility of the generated output sequences, then we can also learn the desired meta-algorithm
- The unique value addition provided by this architecture is that if new CVRP instances are generated with the same number of nodes, vehicle capacity, the same location, and demand distributions as the ones used in the training, then the trained policy will work well. As this framework doesn't require an explicit distance matrix calculation and learns meta-algorithm based on the output sequences, this is suited for constructing routes based on the new data

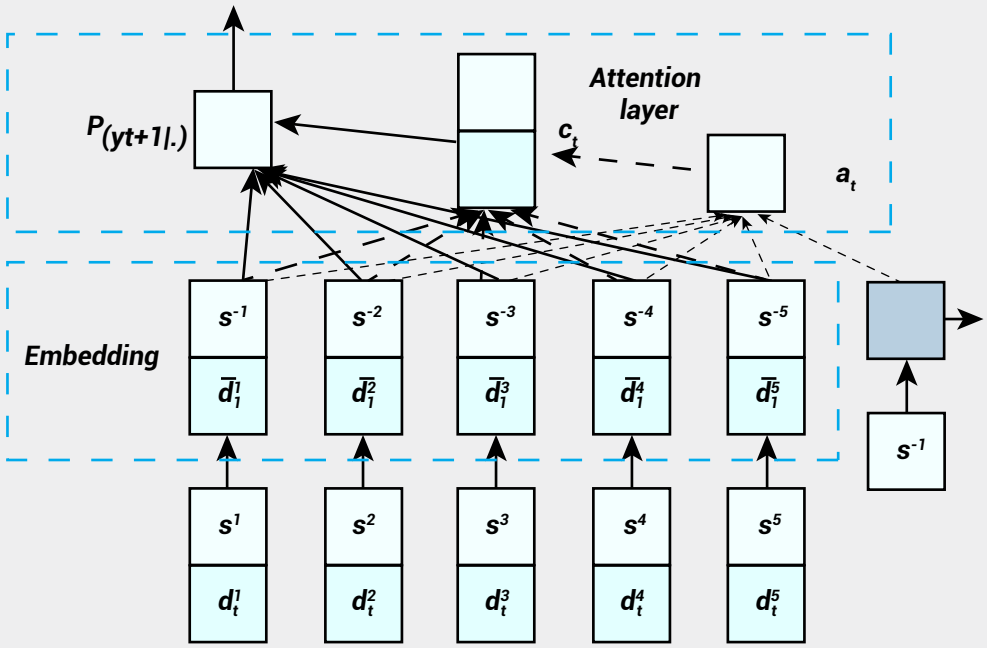

Fig.2: Model invariant of input sequence ordering (Nazari et al. 2018)

iv) Deep Reinforcement Learning (Pang et al., 2020) for CVRP

- The architecture proposed in this work has a dynamic attention model (AM-D) with dynamic encoder and decoder architecture. Each node is characterized dynamically in the context of the graph. Hence, the node information can be adapted to the changes in demand, and the state change of instance can be achieved
- This process is based on encoder-decoder architecture, where the encoder extracts structural features of the input instance, and the decoder incrementally constructs the solution. At each construction step of the decoder, the decoder predicts a distribution over nodes, and one node at a time is selected and appended to the partial solution
- The advancement introduced in this architecture is dynamic encoder-decoder. The encoder and decoder are used alternately to re-construct the embedding of each node and construct a partial solution
- Feature of each node is learned using the embedding learned progressively based on the selected output paths during traversal

3. Setting up the CVRP problem using Deep Reinforcement Learning

There are depot nodes and customer nodes in a graph. A depot node is responsible to fulfil the demands of each customer node and return to its actual state in case of supply over. This instance is shown in the following figure.

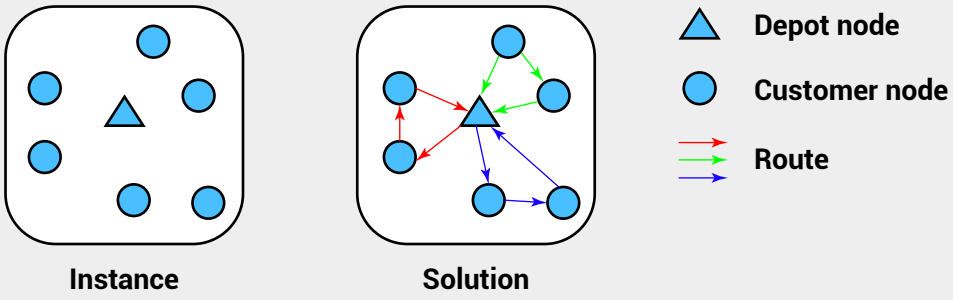

Fig.3: Vehicle Routing Problem (Pang et al., 2019).

Reinforcement learning to solve the CVRP problem: For CVRP, it is required to pick up the best routes during training and augmenting the learning in a current timestamp from the goodness of results produced in the previous timestamps. This move advances using Neural Networks and Reinforcement learning approach the existing problem as a combination that can lead to doing better prediction and decision-making processes. Neural networks can do the automatic feature selection. In the CVRP case, it's about selecting the routes which provide the shortest distances.

In Reinforcement learning-based solutions to CVRP, learning an optimal route is treated as a Markov Decision Process (MDP). An MDP is a discrete-time dynamic system model with three components:

i) S_t-State of the system at time t

ii) a_t-the Action was taken at time t

iii) r_t-the reward at time t

Let S be the set of possible states, A the set of possible actions, and R the set of possible rewards, such that $s_t \in S$, $a_t \in A$, and $r_t \in R$. We assume that A is finite, such that we can choose one of a set of discrete actions at each time point, and that $R = \mathbb{R}$. The reward is meant to represent the value to us (the system controller) of a particular state. The system in Markovian, the current state of the system depends on the state of the system at the previous timestamp and the action taken at the previous timestamp. The transition to the next state is given by:

$$T: P(s_t+1|s_t, a_t)$$

The solution to MDP is to find out the optimal path, where a policy is a function π: S->A, which tells about the action to take in a particular state. The heuristics for making decisions are represented in the form of rules, which can be interpreted as policies to make decisions. These policies are parameterized using neural networks. The idea of learning policy is discussed in the next section.

Learning policies:

i) Policy iteration (Sutton & Barto, 2018): Policy iteration is a dynamic programming-based algorithm that combines *iterative policy evaluation and Policy improvement*. Iterative policy evaluation is an algorithm for evaluating the goodness of a policy by iteratively going through each state and updating expected returns for state-values. Policy improvement is an algorithm that compares outputs retrieved from old policy π and a new one π' and deterministically pick the one with higher expected returns over action values.

ii) Monte Carlo method: Instead of computing every single action-value pair in a state space, giving the true *expectation*, the Monte Carlo sampling method is used to pick up samples of action-value pairs that can yield the best policy for the agent. Sampling the action-value pairs from the state space cuts down the computation cost to a great extent, but it also requires consciously picking those pairs which are optimal. If the samples are finite, the probability of finding optimal pairs also decreases. Similarly, if the number of pairs is increased, the computational cost will rise. The trade-off between computational cost and optimal results is also known as *the exploitation versus exploitation trade-off.* Either exploit a small space, have lesser samples, and thus cost, but it may not be near to the optimal solution; or explore the space more, get more samples, and thus cost and enhance the likelihood of getting a near-optimal solution.

State, Action, Rewards, and Policy:

From a Reinforcement learning standpoint, the following are the definitions of States, Actions, Rewards, and Policy.

i) State: Partial solution of the instance (set of nodes represented as a graph) and the feature of each node of the graph.

ii) Action: Choice of choosing the next node to visit.

iii) Rewards: Negative of total tour length.

iv) Policy: Heuristic strategy parameterized by a neural network.

3.1 Data for CVRP

We have arbitrarily taken 1 depot node and 20 customer nodes. The depot nodes and the customer nodes can be defined by the coordinates in the X-Y plane. The distance between each node can be calculated using Euclidean distance. Table 1 shows the location of 1 depot nodes: Each customer node in a graph has a demand associated.

Table 1. Coordinates of depot nodes and customer nodes and demands of each customer node.

Node No	X Coordinate	Y Coordinate	Demand
Depot	0.848307	0.32357132	
Node1	0.29036105	0.3107828	0.16666667
Node2	0.84289145	0.9391912	0.13333334
Node3	0.7035593	0.43545485	0.06666667
Node4	0.20168412	0.5328256	0.13333334
Node5	0.06545448	0.50693643	0.23333333
Node6	0.53601944	0.5814208	0.1
Node7	0.76553667	0.54667234	0.2
Node8	0.33509946	0.4530629	0.1
Node9	0.29120958	0.8327706	0.1
Node10	0.5238005	0.6762059	0.2
Node11	0.94022834	0.8333516	0.16666667
Node12	0.68327105	0.21968603	0.03333334
Node13	0.392851	0.72069836	0.2
Node14	0.84294224	0.36046684	0.03333334
Node15	0.8818841	0.04076123	0.06666667
Node16	0.6190139	0.31246388	0.23333333
Node17	0.71776164	0.03977752	0.3
Node18	0.9660357	0.4579624	0.23333333
Node19	0.39407074	0.259192	0.03333334
Node20	0.46775055	0.49120617	0.2

Similarly, we have arbitrarily created multiple graphs for training (called graph instance). Each customer node is defined by the position in X, Y coordinates.

These are also randomly assigned demand values between 0-1. The capacity of the vehicle starting from the Depot node is taken as 1, and as a constraint to the CVRP, the demand at any customer node cannot exceed the capacity of the vehicle. So, the demand values range from 0-1.

3.2 The cost function

The loss is the difference between the baseline REINFORCE (Williams, 1992) path length value and the value calculated on the paths obtained by the RL algorithm. The optimal route is the sequence of nodes traversed by the vehicle where the traversal cost is found minimum.

3.3 The Environment of the CVRP agents

As a vehicle starts from a depot node, the customer nodes are being marked as *unvisited*. During the journey, the customer nodes will be marked *visited*. The functions in the environment file are written to ensure that any customer node once visited would be masked and a vehicle can return to either the depot node or any other customer node except the visited nodes.

4. Results

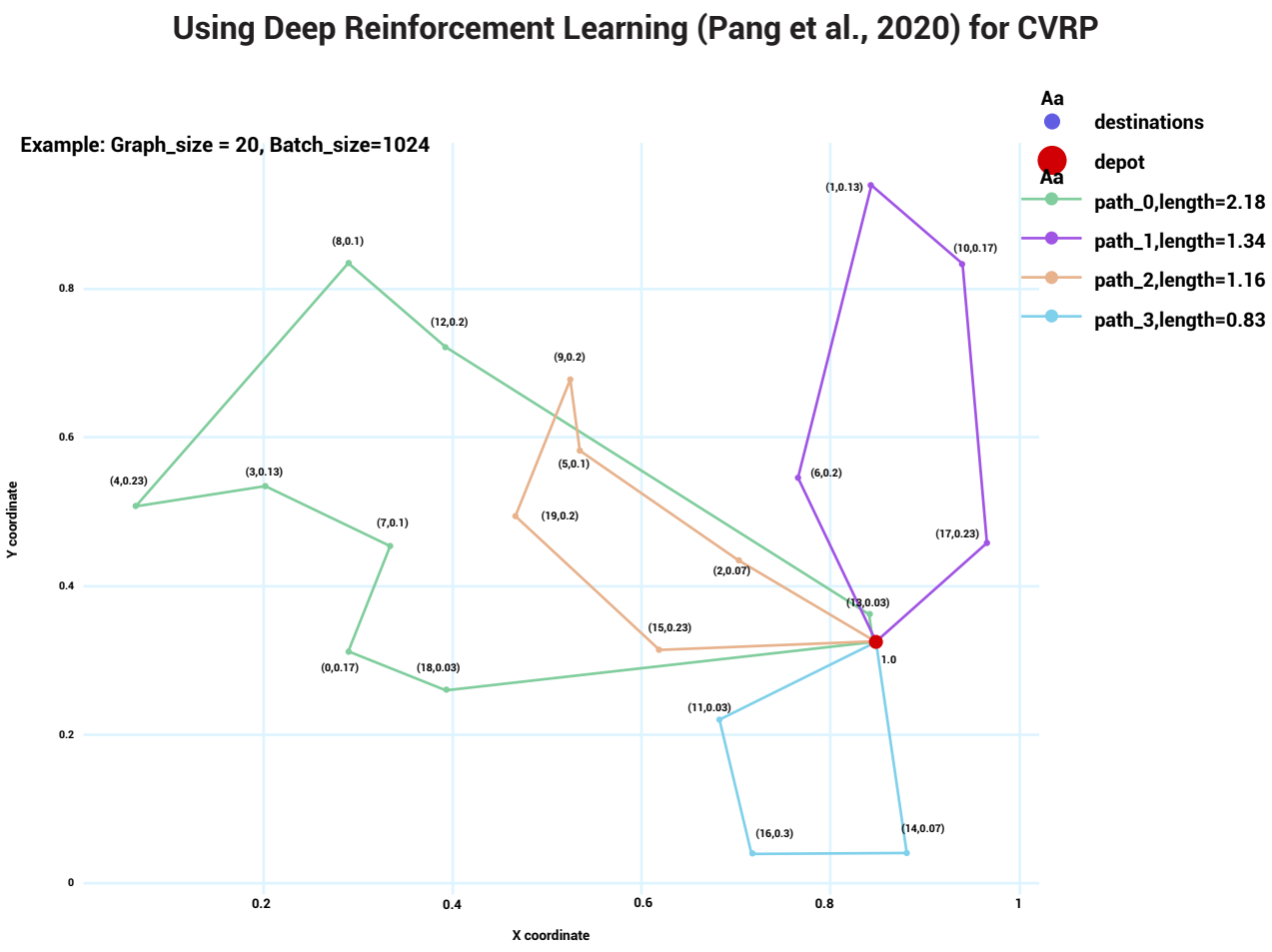

Fig.4: Graph solution for 20 nodes generated by DeepRL

Optimal path: For a CVRP problem, the following is the optimal path obtained for a graph instance and a depot. The cost of the path is 8.83766, which can be further improved by hyperparameter tuning.

Current path: [0.0, 19.0, 1.0, 8.0, 4.0, 5.0, 9.0, 13.0, 14.0, 0.0, 18.0, 11.0, 2.0, 7.0, 0.0, 3.0, 6.0, 10.0, 20.0, 16.0, 0.0, 12.0, 17.0, 15.0, 0.0]

Where 0.0 indicates the depot node. The links between the paths are directed.

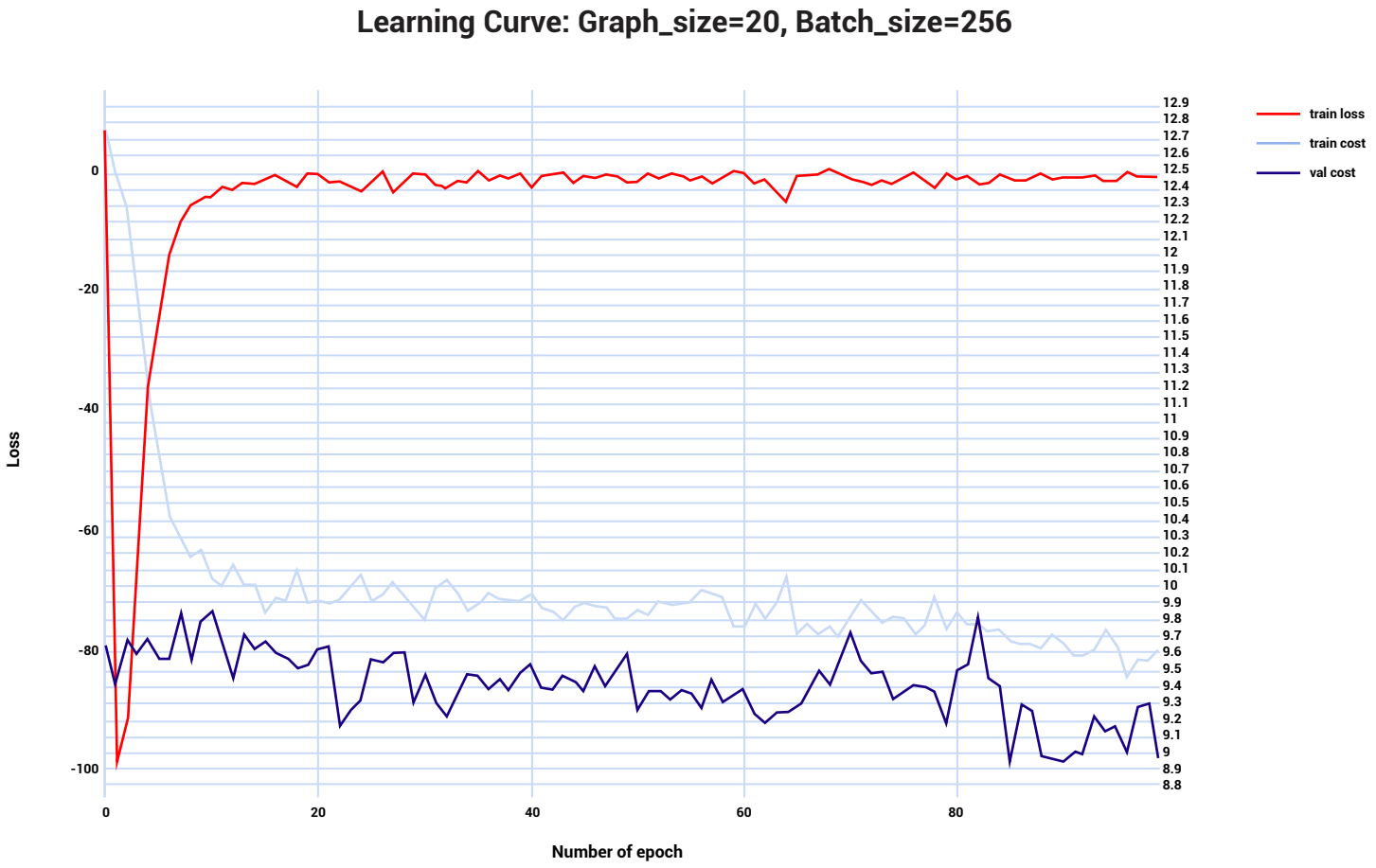

Fig.5: Cost and loss values vs. Epochs

Cost and loss values vs. Epochs: The above plot shows the cost and loss vs. the number of epochs. As the training continues, we can observe that the training loss decreases. This type of result can be further improved by hyperparameter tuning. Similarly, it can be observed that the training cost and validation cost (which is negative of the path length) are decreasing with training.

For comparison purpose, we have used Genetic Algorithm (GA-CVRP):

Genetic Algorithm-CVRP

Genetic Algorithm (GA) is a search-based optimization technique based on the principles of Genetics and Natural Selection. It is used in finding optimal or near-optimal solutions to overcome difficult problems that would take a long time to solve in the usual cases.

Solution Process:

- **Initialization:** In the beginning, an initial generation must be defined. This can be done using a random initialization process.
- **Selection:** As a first step, we select a proportion of the existing population to breed a new generation. The selection is done on a fitness-based approach where fitter individuals are more likely to breed than others.
- **Reproduction:** During the reproduction phase, the next generation is created using the two basic methods, crossover, and mutation. For every new child, a pair of parents is selected from which the child inherits its properties. In the crossover process, the genotype is taken from both parents and combined to create a new child. With a certain probability, the child is further exposed to some mutation, which consists of modifying certain genes. This helps to further explore the solution space and ensure, or preserve, genetic diversity. The occurrence of mutation is generally associated with low probability. A proper balance between genetic quality and diversity is therefore required within the population to support efficient search.

Table 2. Results comparison of "Deep Learning- Reinforcement Learning (DeepRL) approach" and "Genetic Algorithm (GA) approach" with 10 Graphs for 20, 50, 70 and 100 customer nodes.

No of Nodes	Cost (RL-CVRP)	Time Taken (Sec)	Cost(GA-CVRP, Pop=100, Iteration=1000)	Time Taken (Sec)	Cost(GA-CVRP, Pop=100, Iteration=1000)	Time Taken (Sec)
20 Nodes	6.413115	0.13227	7.429141	8.82067308	6.617184412	92.49540444
50 Nodes	13.48943	0.17408	19.92717	79.6166889	17.3899509	830.7374334
70 Nodes	18.00218	0.38383	30.76816	180.8643137	26.60526445	1919.966506
100 Nodes	23.6285	0.58476	44.56011	458.3558464	39.18761302	4683.238334

Note: These numbers are average of 10 Graphs for 20, 50, 70 and 100 nodes.

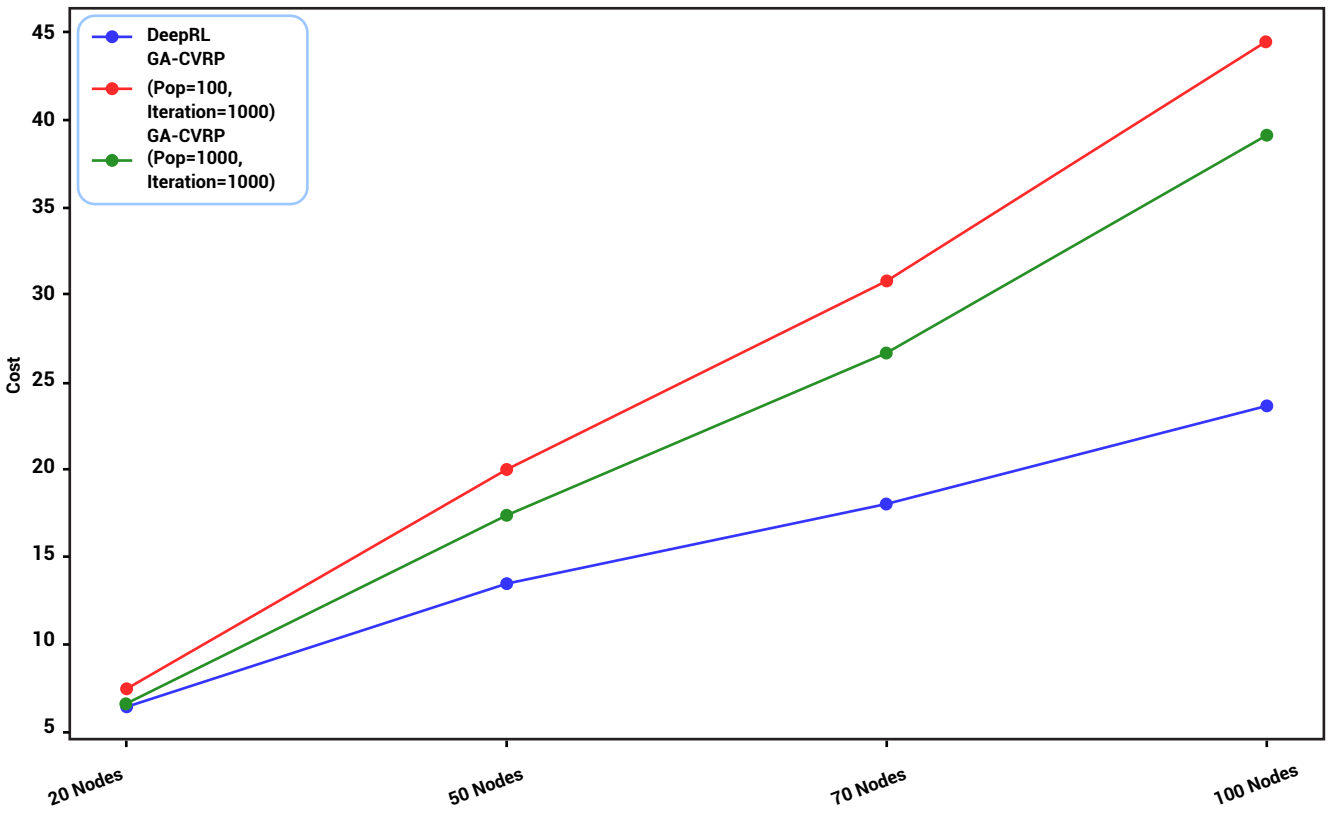

Fig.6: Cost vs. Nodes plot with different Algorithms to solve CVRP

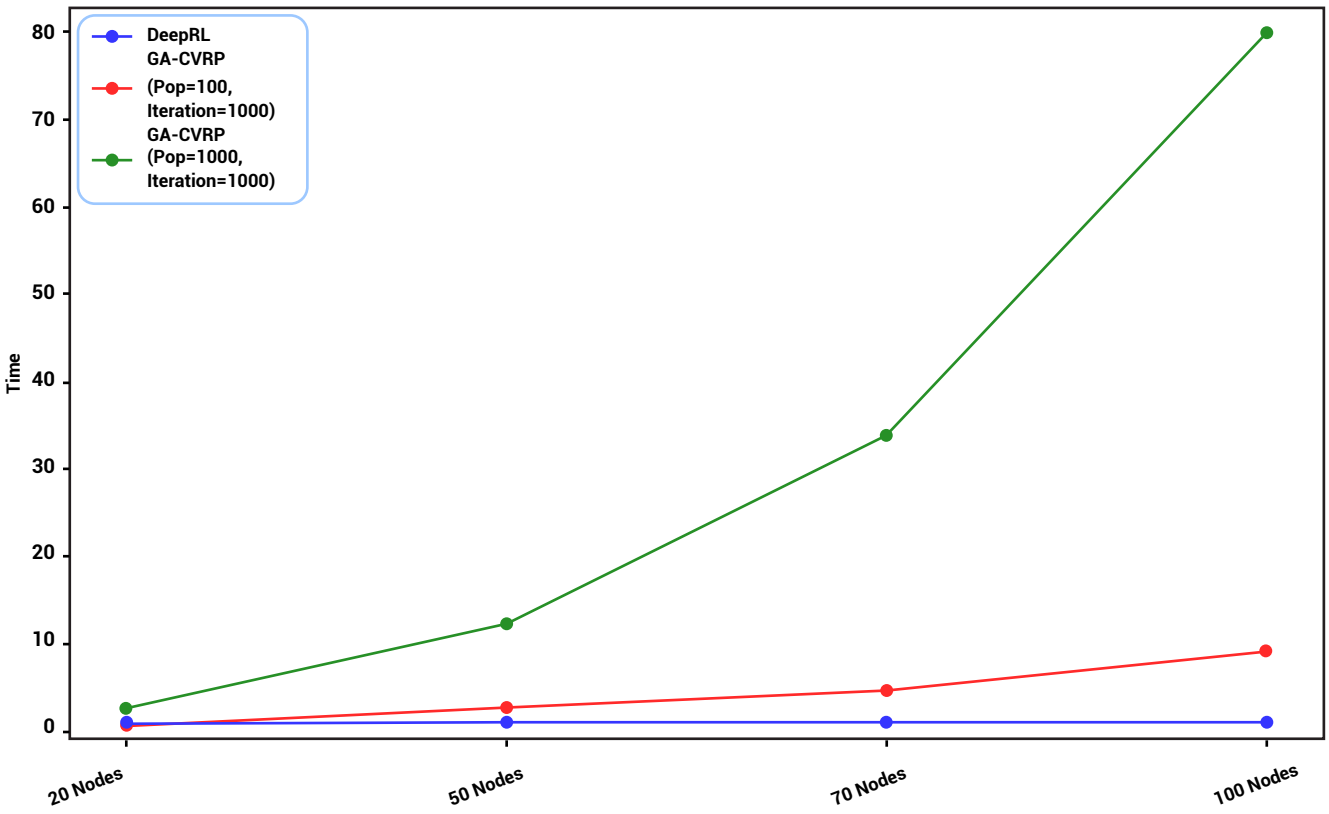

Fig.7: Time vs Nodes plot with different Algorithms to solve CVRP

Report Summary:

1. Time taken by DeepRL is very less and almost remains constant for different graph sizes.
2. Graph generated by DeepRL and GA-CVRP for 20 Nodes has almost the same Cost and Time.
3. Time Taken by GA-CVRP increases significantly with an increase in the number of nodes or an increase in population size (mentioned as POP in figures 6 & 7). It means DeepRL is comparably very efficient when the number of customer nodes increases.
4. Graph generated by DeepRL is more optimized as we increase the number of Nodes when compared with GA-CVRP.

5. Future Work

In this whitepaper, we have used the Deep Reinforcement Learning algorithm to solve the CVRP. There are two important functionalities to be added to the codebase, as mentioned below.

i) Extending “Single depots Multiple customer nodes capacity planning problem” to “Multiple depots Multiple customer nodes capacity planning problem." In the real world, there can be multiple depot nodes, and the solution needs to be optimized considering the possibilities of any depot to any customer nodes with 2 constraints, and they are:

a) The demand of each customer node should not be more than the depot node capacity

b) Each customer node must be traversed only once

ii) Enriching the solution to take customer nodes and depot nodes as geographic locations. This technique will aid to optimize the routes between cities in the real-world with certain constraints. Moreover, using Euclidean distances is may not be the most reliable way to compute the distances between different cities. There are multiple factors like the traffic situation, roadway conditions, etc., that lead to the failure of CVRP using distances as evaluation measures. Next up, we will include “travel time” as a measure to evaluate the performance.

6. References

1] Bello, I., Pham, H., Le, Q. V., Norouzi, M., & Bengio, S. (2016). Neural combinatorial optimization with reinforcement learning.arXiv preprint arXiv:1611.09940.

[2] Vinyals, O., Fortunato, M., & Jaitly, N. (2015). Pointer networks. In Advances in neural information processing systems (pp. 2692-2700).

[3] Nazari, M., Oroojlooy, A., Snyder, L., & Takác, M. (2018). Reinforcement learning for solving the vehicle routing problem. In Advances in Neural Information Processing Systems (pp. 9839-9849).

[4]Peng, B., Wang, J., & Zhang, Z. (2019, November). A Deep Reinforcement Learning Algorithm Using Dynamic Attention Model for Vehicle Routing Problems. In International Symposium on Intelligence Computation and Applications (pp. 636-650). Springer, Singapore.

[6] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning.Machine learning, 8(3-4), 229-256.

[7] Sutton, R. S., & Barto, A. G. (2017). Reinforcement learning: An introduction,(complete draft).

[8] Chunyu REN (2012) Applying Genetic Algorithm for Capacitated Vehicle Routing Problem

[9] Abdul Kadar, Abdul Kadar Muhammad Masum, Md. Faisal Faruque, & Mohammad Shahjalal (2011) Solving the Vehicle Routing Problem using Genetic Algorithm

[10] Rani, K., & Kumar, V. (2014). Solving travelling salesman problem using genetic algorithm based on heuristic crossover and mutation operator. International Journal of Research in Engineering & Technology, 2(2), 27-34.

About Affine

Affine is a Data Science & AI Service Provider, offering capabilities across the analytical value chain from data engineering to analytical modeling and business intelligence to solve strategic & day-to-day business challenges of organizations worldwide. Affine is a strategic analytics partner to medium and large-sized organizations (majorly Fortune 500 & Global 1000) around the globe that creates cutting-edge creative solutions for their business challenges.

Talk to our decision science analytical experts

Discover how Affine can support your retail transformation journey.

www.affineanalytics.com | info@affineanalytics.com