

Overview and Experimental Study of Learning-based Optimization Algorithms for the Vehicle Routing Problem

Bingjie Li, Guohua Wu, Yongming He, Mingfeng Fan, and Witold Pedrycz *Fellow, IEEE*,

Abstract—Vehicle routing problem (VRP) is a typical discrete combinatorial optimization problem, and many models and algorithms have been proposed to solve the VRP and its variants. Although existing approaches have contributed a lot to the development of this field, these approaches either are limited in problem size or need manual intervening in choosing parameters. To solve these difficulties, many studies have considered the learning-based optimization (LBO) algorithms to solve the VRP. This paper reviews recent advances in this field and divides relevant approaches into *end-to-end approaches* and *step-by-step approaches*. We performed a statistical analysis of the reviewed articles from various aspects and designed three experiments to evaluate the performance of four representative LBO algorithms. Finally, we conclude the applicable types of problems for different LBO algorithms and suggest directions in which researchers can improve LBO algorithms.

Index Terms—Vehicle routing problem, Learning-based optimization algorithms, Reinforcement learning, End-to-end approaches, Step-by-step approaches.

I. INTRODUCTION

THE VRP is one of the most widely researched problems in transportation and operations research areas [1]. The canonical VRP has a simple structure and can be seen as a basic discrete combinatorial optimization problem. Many optimization problems can be transformed into the form of the VRP. Hence, the VRP can be used to demonstrate the optimization performance of different algorithms, which can help to design efficient algorithms. Moreover, the VRP has significant practicality in the real world. VRP variants have appeared in many fields [2, 3], particularly in the logistic industry that is flourishing with the development of the globalization. Effective routing optimization can save a lot of cost (generally ranging from 5% to 20%) [4], which is of great significance for improving distribution efficiency [1, 5–8] and increasing economic benefits of enterprises [9–11]. The VRP is still continuously attracting attention from researchers [12, 13].

Bingjie Li Guohua Wu and Mingfeng Fan are with the School of Traffic and Transportation Engineering, Central South University, Changsha 410075, China. E-mail: csulbj@csu.edu.cn; guohuawu@csu.edu.cn; mingfan@csu.edu.cn.

Yongming He is with the College of Systems Engineering, National University of Defense Technology, Changsha 410073, China (e-mail: heyongming10@hotmail.com)

Witold Pedrycz is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 2V4, Canada, with the Department of Electrical and Computer Engineering, Faculty of Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia, and also with the Systems Research Institute, Polish Academy of Sciences, Warsaw 01447, Poland. E-mail: wpedrycz@ualberta.ca.

However, it is still an challenging optimization task due to the NP-hard characteristic [14] and its different complex variants.

Optimization algorithms for solving the VRP can be crudely divided into three categories: exact, heuristic, and the LBO algorithms. Fig. 1 shows the overview of referred algorithms. Heuristic algorithms and exact algorithms as traditional algorithms for combinatorial optimization problems, many literatures have reviewed their applications in the VRP [15]. Mańdziuk [16] reviewed papers of using heuristic algorithms to solve the VRP between 2015 and 2017. Adewumi and Adeleke [17] emphasized recent advances of traditional algorithms in 2018, while Dixit *et al.* [18] reviewed some of the recent advancements of meta-heuristic techniques in solving the VRP with time windows (VRPTW). Exact and heuristic algorithms have made significant progress in the VRP, but the recent research trend is to design a flexible model that can solve the VRP with large scale and complex constraints more quickly. Exact algorithms need adequate time to get an optimal solution when solving a VRP with large scale [19]. The optimality of heuristic algorithms cannot be guaranteed and their computational complexity is not satisfactory [20]. What's more, these two kinds of algorithms usually need specific design for solving the concrete problem. In recent years, the LBO algorithms have been applied to different combinatorial optimization problems [23, 24], and many remarkable research breakthroughs have been achieved. Hence, the LBO algorithms have attracted significant attention in the field of the VRP.

The LBO algorithms can learn a model from training sets to obtain optimization strategies [21, 22], which could automatically produce solutions of online tasks by end-to-end or step-by-step approaches. In end-to-end approaches, the model is trained to approximate a mapping function between the input and the solution, and the model can directly output a feasible solution when given an unknown task in application. While step-by-step approaches learn optimization strategies that could iteratively improve a solution rather than outputting a final solution directly. Both approaches have strong learning ability and generalization. They can overcome the deficiencies of the tedious parameter tuning of exact and heuristic algorithms and rapidly solve online instances through the advantage of offline training. However, the LBO algorithms still have some technological bottlenecks to be settled, such as training data limitation, generality limitation, and so on.

Overall, although the application of the LBO algorithms to the VRP has become popular and significant achievements have been gotten [25, 26] in recent years, up to now there

are no comprehensive assessment experiments and in-depth analyses on the characteristics of different LBO algorithms in solving the VRP. Therefore, we aim to present an elaborate overview and experimental study of related work to fill the gap in this field. Contributions of the paper are mainly on the following aspects:

- The paper briefly introduces the applications of the LBO algorithms to the VRP to aid beginners in understanding the development of this field.
- The paper discusses the advantages and disadvantages of different LBO algorithms based on extensive experiments on different datasets.
- The paper suggests promising research trends on using the LBO algorithms to solve the VRP in the future.

The remainder of this paper is structured as follows. In Section II, we introduce the background of the VRP, including the mathematical model and classical optimization algorithms. In Section III, we review the related literatures and provide a summative analysis of the references. In Section IV, the experimental comparisons among different algorithms are presented. The final section summarizes the full text and suggests several research trends of using the LBO algorithms to solve the VRP.

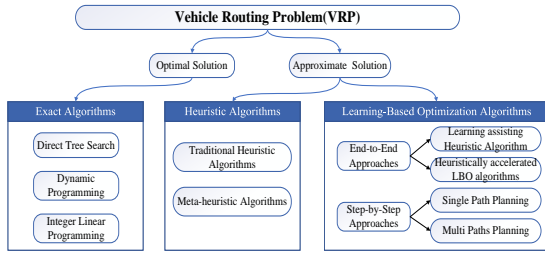


Fig. 1. Overall classification of the referred algorithms of the VRP.

II. BACKGROUND OF THE VRP

As the routing optimization problems are increasingly crucial in industrial engineering, logistics, etc, a series of research breakthroughs have been achieved in the VRP and many algorithms have been proposed. In this section, we briefly introduce the background of the VRP.

A. Variants of the VRP

The mathematical model of the VRP is first proposed by Dantzig and Ramser [27] in 1959. A few years later, Clarke and Wright [28] proposed an effective greedy algorithm to solve the VRP, which also launched a research boom in the VRP. The VRP can be defined as follows: for a series of nodes with different demands, the aim is to plan routes with the lowest total cost under various side constraints [29, 30] for vehicles to serve these nodes sequentially according to the planning routes. With its wide application in the real world, the VRP must consider more constraints, which results in the derivation of many variants. Fig. 2 presents the hierarchy of VRP variants.

The basic version of the VRP is the capacitated vehicle routing problem (CVRP) [16, 31]. In the CVRP, each customer is served only once and the sum of goods carried by a vehicle

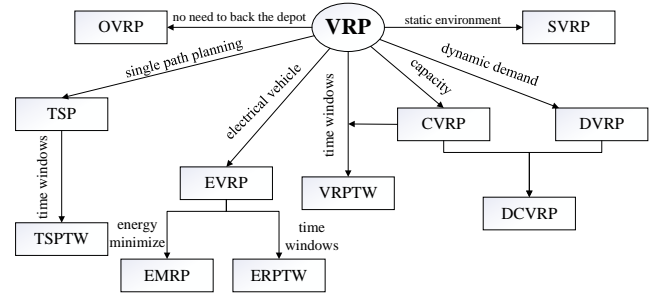


Fig. 2. Different variants of the VRP.

should not exceed its capacity [32]. This type of problem has the following characteristics: (1) vehicles leave from the depot and return to it; (2) vehicles are permitted to visit each customer on a set of routes exactly once; (3) each customer has a non-negative demand of delivery. Fig. 3 is the schematic diagram of the CVRP.

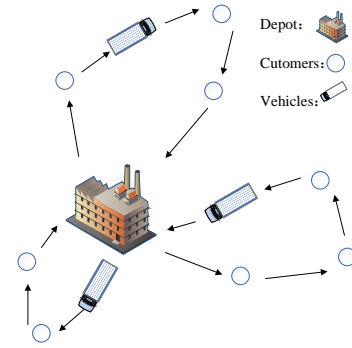


Fig. 3. Schematic diagram of the CVRP

B. Mathematical model of the VRP

The VRP can be described by an integer programming model [19], and we mainly present the mathematical model of the CVRP in this subsection. According to [33], the mathematical model of the CVRP can be expressed as follows, where N represents the set of customers, the number 0 represents the depot, K represents the number of vehicles, A represents the set of arcs and V represents the set of nodes, with $V = N \cup \{0\}$. We define a binary decision variable $x_{i,j}^k$ to be 1 if the pair of nodes i and j are adjacent in the route of the vehicle k ; otherwise, $x_{i,j}^k$ equals 0. Each vehicle's maximum capacity is C . u_i^k represents the surplus capacity of the vehicle k after it serves the customer i , and q_j indicates the demand of customer j .

$$\text{minimize } \sum_{k \in K} \sum_{i,j \in A} c_{i,j} x_{i,j}^k, \quad (1)$$

$$\text{s.t. } \sum_{k \in K} \sum_{j \in V, j \neq i} x_{i,j}^k = 1, i \in N \quad (2)$$

$$\sum_{k \in K} \sum_{i \in V, i \neq j} x_{i,j}^k = 1, j \in N \quad (3)$$

$$\sum_{k \in K} \sum_{i \in N} x_{i,0}^k = K \quad (4)$$

$$\sum_{k \in K} \sum_{j \in N} x_{0,j}^k = K \quad (5)$$

$$u_j^k = \begin{cases} u_i^k - q_j, & \text{if } x_{i,j}^k = 1, \{i,j\} \in A : j \neq 0 \\ C, & \text{if } j = 0 \end{cases} \quad (6)$$

$$q_j \leq u_i^k \leq C, \text{ if } x_{i,j}^k = 1 : i \in V, j \in N \quad (7)$$

$$x_{i,j}^k \in \{0, 1\}, \{i,j\} \in A \quad (8)$$

In this formulation, the objective is to minimize the total cost of transportation and $c_{i,j}$ represents the cost of arc $\{i,j\}$. Equations (2) and (3) indicate that each customer can be served only once. Constraints (4) and (5) represent that both number of vehicles leaving the depot and returning to the depot should be equal to the total number of vehicles, which means that each vehicle should start from and end at the depot. Equation (6) shows the update process of the surplus capacity of the vehicle k , and constraint (7) represents that the surplus capacity of the vehicle must not be less than the demand of its next customer j while does not exceed the maximum capacity C . Constraint (8) is used to ensure decision variable $x_{i,j}^k$ is a binary.

C. Traditional Optimization Algorithms for the VRP

Traditional optimization algorithms for solving the VRP include exact algorithms and heuristic algorithms. Fisher [34] divided the development of traditional algorithms applied to the VRP into three stages. From 1959 to 1970 is the first stage, and simple heuristic algorithms were mainly used in this stage, such as the local heuristic and greedy algorithms; the second stage is from 1970 to 1980, and this stage primarily applied exact algorithms; after 1980 is the third stage and meta-heuristic algorithms began to be applied to VRPs. These traditional algorithms have been successfully used to solve different VRP variants[35].

1) *Exact algorithms*: According to [36], exact algorithms for solving the VRP can be divided into three categories: direct tree search (such as the branch-and-bound algorithm [37]), dynamic programming [38], and integer linear programming (such as the column generation algorithm [39]). Many breakthroughs were achieved using exact algorithms in the 1970s. Christofides and Eilon [40] studied the CVRP and dynamic VRP (DVRP) by designing three approaches: a branch-and-bound approach, a saving approach, and a 3-optimal tour method. Christofides *et al.* [41] used tree search algorithms to solve the VRP. Although these exact algorithms are easy to understand, their computational expense is prohibitive in large-scale problems [42, 43].

2) *Heuristic algorithms*: Heuristic algorithms can be divided into traditional heuristic algorithms and meta-heuristic algorithms. Referring to [44], traditional heuristic algorithms can be divided into constructive and two-stage heuristic algorithms. The former gradually generates feasible solutions to minimize costs (e.g. the savings algorithm [28]). The latter first clusters nodes and then constructs multiple feasible routes to satisfy constraints. The quality of the solution is improved

by changing the positions of nodes between or within routes (e.g. [45]). Constructing the neighborhood of a solution is a significant part of heuristic algorithms. However, traditional heuristic algorithms perform an entire search without focus; therefore, they only explore neighborhoods in a limited manner. A meta-heuristic algorithm can focus on potential areas by combining intelligently complex search rules [46] and memory structures. As reported in [47], meta-heuristic algorithms generally produce better solutions than traditional heuristic algorithms (typically between 3% and 7%).

Although exact and heuristic algorithms have been developed for many years, they all have their own limitations. Both algorithms frequently require optimization expertise to model the problem and design effective search rules. Moreover, algorithms based on search frequently encounter the challenge of a trade-off between searching efficiency and solution accuracy when solving complex combinatorial optimization problems. This is apparent because finding an optimal solution comes at the expense of searching for a larger neighborhood and longer computational time. With the development of computer technology since 2010, the LBO algorithms have become a popular research topic and many breakthroughs have been obtained in using them to solve optimization problems. Along with the deepening of research on the VRP, it is necessary to design algorithms that can more rapidly and efficiently solve problems. Hence, the LBO algorithms are beginning to be applied to solve the VRP.

III. LEARNING-BASED OPTIMIZATION ALGORITHMS FOR THE VRP

From a technical point of view, the LBO algorithms usually include three kinds of learning modes [48, 49]. If the agent learns on data with labels, this training mode is called supervised learning (SL). In contrast, learning from unlabeled data is named unsupervised learning (UL). Note that UL is commonly used for the parameter optimization of continuous problems; therefore, it is rarely used in the literatures reviewed in our paper. The final type of learning framework is reinforcement learning (RL), which requires the agent to learn from interaction with the environment. RL involves agents sense the environment and learn to select optimal actions through trial and error [50, 51]. Compared with traditional optimization algorithms, the LBO algorithms have three advantages:

- The LBO algorithms do not require substantial domain knowledge [52] for mathematical modeling and parameter tuning, which enables the LBO algorithms to model routing problems in a real-world more flexibly.
- The LBO algorithms can automatically construct an empirical formula to approximate the mapping function between the inputs and solutions through pre-training on a dataset.
- The LBO algorithms can automatically extract optimization knowledge from training data and give computers the ability to learn without being explicitly programmed [53].

The ability of autonomous and offline learning results in LBO models not requiring any hand-engineered reasoning and

rapidly providing a promising solution for unknown data [52]. The LBO algorithms have been applied to different fields [22], such as video games [54], Go [55], robotics control [56] and image identification [57]. Many studies have used the LBO algorithms for the VRP, and we divide related frameworks into two types: end-to-end and step-by-step approaches.

A. Step-by-step approaches

Step-by-step approaches learn optimization strategies that could iteratively improve a solution rather than outputting a final solution directly. These approaches can find a promising solution, but they have low time efficiency for training since their search space is more extensive. We subdivide this type of approaches into learning assisting heuristic and heuristically accelerated LBO algorithms according to different solving frameworks.

1) *Learning assisting heuristic algorithms*: Heuristic algorithms use a series of operators to find the optimal solution [59]. During search process, heuristic algorithms would generate considerable information about how to evolve and search in different stages [58], but this useful information is not comprehensively and well utilized by heuristic algorithms. Hence, many studies have used the LBO algorithms as assistants to learn from this information and define the optimal setting of heuristic algorithms. Hence, we call algorithms of these studies *learning assisting heuristic algorithms* because they still search based on heuristic framework.

In their book in 1995, Gambardella *et al.* [60] used Q-learning to construct the initial path for the ant system (AS). The LBO models in both papers of Lima *et al.* [61] and Alipour *et al.* [62] were also used as constructors of initial solutions for heuristic algorithms. Subsequently, many scholars used the LBO models to assist heuristic algorithms from different aspects. Liu *et al.* [63] proposed an improved generic algorithm (GA) with RL called RMGA. They used RL model to determine whether to break the connection relation of the initial tour and used the GA to reselect the next city for the current city. Later, Phiboonbanakit *et al.* [64] used a transfer learning algorithm to aid the GA in dividing customers into regional clusters beforehand. Ding *et al.* [65] used the LBO models to predict the values of nodes for accelerating convergence of solvers, and recently, Sun *et al.* [66] used the LBO model to quantify the likelihood of each edge belonging to an optimal route by the support vector machine (SVM). The LBO models can reduce the search space to simplify problems for heuristics, and all experiments demonstrated that these learning assisting heuristic approaches can significantly improve the performance of original heuristic algorithms and speed up the solving process.

In addition to using the LBO algorithms to assist heuristic algorithms during constructing solution, the LBO algorithms can set parameters of heuristic algorithms. In 2017, Cooray and Rupasinghe [67] used the LBO model to set the mutation rate of the GA for the energy-minimized VRP (EMVRP). They experimented with GA with different mutation rates on instances and selected the optimal parameter settings that can produce minimal energy consumption. It has been proven

that parameter tuning according to data characteristics has a significant effect on improving the applicability of the GA. Al-Duoli *et al.* [68] also used the LBO model to define the parameters values of a heuristic algorithm. Moreover, they used association rules trained using UL to provide an initial solution for the search.

The large neighborhood search (LNS) algorithm is a new heuristic algorithm that was first proposed by Shaw *et al.* [69] in 1997. Compared with previous heuristics, this algorithm based on the ruin-and-recreate principle [70] would explore more complex neighborhoods. The LNS algorithms have outstanding performance in solving various transportation and scheduling problems [71]. Therefore, some scholars have considered using the LBO algorithms to improve the LNS algorithms. The learned LNS algorithm for the VRP was first used by Hottung and Tierney [72] for the CVRP and split delivery vehicle routing problem (SDVRP) in 2019. They adopted a LBO model as the repair operator for the LNS algorithm. Their model is limited by the number of destroyed fragments; hence, it ineffectively solves large-scale problems. Later, both Chen *et al.* [73] and Gao *et al.* [74] used a LBO model as a destroy operator of the LNS algorithm to solve the VRP. Chen *et al.* [73] used proximate policy optimization (PPO) algorithm to train a hierarchical recursive graph convolution network (GCN) as the destroy operator. Then, they simply inserted nodes removed by the destroy operator into the infeasible solution according to the principle of minimum cost. Experiments on Solomon benchmarks [75] and synthetic datasets demonstrated that the model of Chen *et al.* [73] performed better than the adaptive LNS (ALNS) algorithm in terms of the solution quality and computational efficiency. Gao *et al.* [74] considered the effect of graph topology on the solution, and they used an element-wise graph attention network with edge-embedding (EGATE) as an encoder in which both the node set and arc set contribute to the attention weights. They used a principle similar to that in [73] to repair the destroyed solution. Overall, using a LBO model as a destroy operator in the LNS algorithm is more beneficial for generating potential neighbor solutions than as a repair operator.

2) *Heuristically accelerated LBO algorithms*: Although using the LBO models as assistants can effectively improve the performance of heuristic algorithms, these heuristics still need to elicit knowledge from experts and encode it into the program, which is far from being easy in any applications. Hence, some studies proposed to replace knowledge-modeling in conventional algorithms and used heuristics as assistants of the LBO algorithms to search for solutions. We call these approaches *heuristically accelerated LBO algorithms*.

Reinaldo *et al.* [76] proposed a heuristically accelerated distributed deep Q-learning (HADQL) algorithm and compared their model with the ant colony optimization (ACO) algorithm and deep Q network (DQN) on TSP instances. The results indicated that the convergence of their model was fast, but their model required more computation time. Yang *et al.* [77] used the LBO algorithm to approximate the dynamic programming function, and their algorithm outperformed other well-known approximation algorithms on large-scale TSPs. Joe and Lau

[78] used the LBO algorithm to compute the serving cost of each node, then they applied a simulated annealing (SA) algorithm to plan routes to minimize the total cost of the dynamic VRP with time windows and random requirements. Later, Delarue *et al.* [79] also used a LBO algorithm to compute the cost of nodes beforehand and their experiments demonstrated that their model achieved an average gap against the well-known solver OR-Tools of 1.7% on CVRPs. Yao *et al.* [80] considered a real-life route planning problem. They regarded the safety of the planned route as one of the objectives and used RL to train the LBO model. They compared their model with EMLS and NSGA-II on a map of New York. Although their framework has good efficiency and optimality, it requires a large amount of time during training.

Chen and Tian [81] proposed a LBO model with 2-opt algorithm as the search operator. The model selects a fragment of a solution to be improved according to the region-picking policy and uses the rule-picking policy to select a rewriting rule applicable to the region. Both policies are learnt by the LBO model, and the solution is iteratively rewritten continuously until it converges. Because this framework partially improve the solution, it is significantly affected by the initial solution. Lu *et al.* [82] proposed an RL model incorporating several heuristic operators, called the L2I. If the cost reduction after improvement by heuristic operators does not reach the threshold, a random perturbation operator is applied and the operation begins again. Generally, [81] and [82] are similar because both include improvement and breaking during search process. Although they both obtained good experimental results, [82] generated more potential solutions than [81] because [82] designed a rich set of improvement and perturbation operators. Similar to [81], Costa *et al.* [83] and Wu *et al.* [84] used an LBO model as a policy network for the 2-opt operator. [83] built an encoder-decoder framework based on the pointing mechanism [85]. Their greatest contribution was that they used different neural networks (NN) to embed node and edge information respectively, and proved that the consideration of edges has an important effect on solving the VRP. The difference in [84] from [81] is that they did not construct another LBO model to decide the rewritten region, but [84] embed the nodes based on the self-attention mechanism before selecting node pairs to apply 2-opt heuristic. Vlastelica *et al.* [86] turned solvers into a component of the LBO model to search the optimal solution. They used the LBO model to embed a sequence of nodes into a matrix of pairwise distances to input the solver, then they used the gap between the output of the solver and data label as a loss to optimize the LBO model. Later, Ma *et al.* [87] modified the model of Wu *et al.* [84] by separating sequence embedding of the current solution from node embedding. Experimental results showed that their LBO model can capture the circularity and symmetry of VRP solutions more effectively. Although above models can obtain better solutions for complex VRPs by making the use of extensiveness of heuristic search, but they need a longer time to search.

In addition to incorporating heuristics as components of the LBO models, scholars have recently proposed to use the LBO algorithm to automatically select or generate heuristics,

which can be considered as a type of hyper-heuristic algorithm. Hyper-heuristic algorithms have been used to solve the VRP in many studies [88–90]. This kind of algorithms does not search randomly but is guided by other high-level algorithms. Meignan *et al.* [91] first used a multi-agent modular model based on the agent metaheuristic framework (AMF) to build a selective hyper-heuristic framework for the CVRP and SDVRP. In their model, each agent builds an AMF model and selects the low-level heuristics. RL and SL mechanisms are used by agents to learn from experience and other agents respectively. Later, Asta and Özcan [92] designed a hyper-heuristic algorithm based on apprenticeship learning to produce a new heuristic method for the VRPTW. The same LBO algorithm was used by Tyasnurita *et al.* [93] to train the time delay neural network (TDNN) to solve the open vehicle routing problem (OVRP), which does not require vehicles to return to the depot. Moreover, some studies used the LBO algorithms to select heuristics. Both Kerschke *et al.* [94] and Zhao *et al.* [95] trained the LBO model as a selector to select the best algorithms for a given TSP instance. Rodríguez *et al.* [96] constructed a multi-layer perceptron (MLP) to select the best meta-heuristic for solving the VRPTW, and Martin *et al.* [97] used LBO as a selector to select the best parameter settings for randomized Clarke Wright savings for CVRP. Using LBO models as selector of heuristics can leverage complementarity within a set of heuristics to achieve better performance.

B. End-to-end approaches

End-to-end approaches learn a model to approximate a mapping function between the input (the features of the problem) and the output (the solution of the problem), and the model can directly output a feasible solution on unseen test tasks. The time efficiency for training end-to-end approaches is generally better compared to step-by-step approaches, but they currently encounter challenges in finding high quality solutions and scalability. According to the characteristic (whether the vehicle visits the depot multiple times) of problems which the approach is applied to, end-to-end approaches can be divided into single-path and multi-path planning approaches.

1) *Single-path planning*: In some VRP variants, the vehicle is not required to return to the depot, such as the TSP, and the LBO algorithms applied to this type of VRPs output a single path that connect all of nodes. Hence, we refer to these LBO algorithms as *single-path planning approaches*.

Standard NNs rely on the assumption of independence among data points and fixed-length inputs and outputs, which is unacceptable for sequence-to-sequence problems [99]. States of sequence-to-sequence problems are related in time or space, such as words from sentences and frames from a video. To overcome this shortcoming, recurrent neural networks (RNNs) was designed [100], and other RNN architectures, *long short-term memory* (LSTM) [101] and *bidirectional recurrent neural networks* (BRNNs) [102] were also introduced for sequence learning in 1997. Later, many studies used these architectures in different sequence-to-sequence problems, such as natural language translation [103] and image captioning

[104]. However, these models compress all the information of the encoder into a fixed-length context vector, which prevents the decoder from focusing on more important information when decoding. Hence, Vinyals *et al.* [85] introduced the attention mechanism [105] into the sequence model of [103] to enable the decoder to focus on important embeddings and called their model *Pointer Net*. The attention mechanism enables the LBO model to select the contexts that are most closely related to the current state as input. They trained an RNN with a non-parametric softmax layer using SL to predict the sequence of the cities. Although their end-to-end model can directly output the target point of the next step, it is not effective for large-scale TSP. The experimental results also demonstrated that *Pointer Net* produced better solutions when the number of nodes was less than 30, but performed poorly on TSP40 and TSP50.

Vinyals *et al.* [85] first proposed an end-to-end model for solving combinatorial optimization problems, but their research needs further improvement. Because SL is undesirable for NP-hard problems, Bello *et al.* [106] combined the RL algorithm with *Pointer Net* to solve the TSP in 2016. They proved that their method was superior to that of Vinyals *et al.* [85] on TSP100. Levy and Wolf [107] transformed *Pointer Net* to solve other sequence problems in addition to the TSP. They inputted two sequences to two *Pointer Nets* and used a convolution neural network (CNN) to output the alignment scores, which were subsequently converted into distribution vectors using a softmax layer. Later, Deudon *et al.* [108] used multiple attention layers and a feedforward layer as encoder to simplify *Pointer Net*. Some studies have also extended the structure of *Pointer Net* to solve other single-path problems. Li *et al.* [109] used *Pointer Net* to solve the multi-objective TSP (MOTSP). They first decomposed the multi-objective problem into a series of sub-problems and then used *Pointer Net* to solve each sub-problem. All sub-problems can be solved sequentially by transferring the network weights using the neighborhood-based parameter-transfer strategy. Kaempfer and Wolf [110] added a leave-one-out pooling to extend *Pointer Net* to solve the multiple TSP (MTSP), and Le *et al.* [111] trained *Pointer Net* using RL to plan a route for cleaning robots. Ma *et al.* [112] extended *Pointer Net* to solve the TSP. [112] used a graph neural network (GNN) to encode distances of cities as context vectors of the attention mechanism, which is beneficial for solving a large-scale TSP. They also improved their model to a hierarchical framework to solve the TSP with time windows (TSPTW), in which they solved the vanilla TSP in the first level and solved the constraint of time windows in the high level.

Although *Pointer Net* has been widely accepted to solve the TSP since its pioneer contribution, many scholars have proposed other end-to-end models to solve the TSP. As stated in [113], the mentioned neural architectures of *Pointer Net* cannot yet effectively reflect the graph structure of the TSP; therefore, Dai *et al.* [113] proposed incorporating a GNN to construct solutions incrementally for the TSP, which is named as *Structure2vec(S2V)* [114], and the embeddings of their model are updated continuously by adding new nodes into currently infeasible solutions. Later, Ottoni *et al.* [115] used a

set of statistical techniques called the response surface model (RSM) to determine values of the learning rate and discount factor of the LBO algorithm. Although the experiments proved that using RSM can significantly improve the performance of the LBO model, their model need to calculate optimal parameters per instance which limited the generalization of the model. To solve the large-scale TSPs (up to 10000 nodes) in an acceptable runtime, Fu *et al.* [116] innovatively introduced a graphic transformation and heat map technique into an end-to-end model. They used graph sampling to abstract sub-graphs from the initial large graph, and the trained LBO model output the corresponding heat map (probability matrix over the edges). Finally, all the heat maps were merged into the final heat map, and the RL algorithm was used to output the optimal solution. The only limitation is that the model only experimented on simulation data but not on other benchmarks or real data. Zhang *et al.* [117] proposed an LBO approach for a kind of TSP that allows the vehicle to rejects orders, which was called the TSPTWR. They modified the model of Kool *et al.* [118] to output an initial solution for the TSP and then used a greedy algorithm to post-process the solution by rejecting the nodes that violate time windows. Compared with tabu search (TS), this method has a shorter solving time and better results. Because decisions of end-to-end approaches cannot reverse, Xing *et al.* [119] combined a GNN and Monte Carlo tree search (MCTS) to solve the TSP. In contrast to previous LBO models that directly made a decision by using a prior probability generated by training, they used MCTS to further improve the decision. Although their model outperforms previous LBO models in test sets of any size, the solving time is much longer.

Groshev *et al.* [120] and Joshi *et al.* [121] trained a GCN by SL to solve the TSP. What's more, [120] further expanded their model by using a trained GCN to guide heuristic algorithms to output solutions and then they used these solutions as labels to retrain the GCN for large-scale TSPs. Prates *et al.* [122] also used SL to train an LBO model. They considered the edge weights as features of per instance, and the deviation of their solutions from the optimal can be less than 2%. Previous LBO models are usually trained and tested on Euclidean space, which makes these models unavailable in instances where nodes are not uniformly distributed. Consequently, Sultana *et al.* [123] tested their model in a non-Euclidean space. Although above LBO models trained by SL require fewer samples compared with those LBO models trained by RL, the optimal solutions that are selected as labels have a significant influence on the performance of the models. Joshi *et al.* [124] performed controlled experiments between the RL model [118] and SL model [121], and they also proved that RL model has better generalization.

2) *Multi-path planning approaches*: Most routing problems are limited by different constraints due to the complex environment. In these VRPs, the LBO algorithms need to output multiple path loops since vehicles need return to the depot more than once. We termed these LBO algorithms as *multi-path planning approaches*.

To the best of our knowledge, Nazari *et al.* [125] first proposed an end-to-end approach to solve the CVRP. They

believed that the order of input is meaningless; thus, they expanded *Pointer Net* by utilizing element-wise projections to map static elements (coordinates of nodes) and dynamic elements (demands of nodes) into a high-dimensional input. The dynamic input is directly fed into the attention mechanism and not complexly computed using an RNN; thus, their model is easier to update embeddings during constructing the solution compared with *Pointer Net*. The optimality of the model was proved by Ibrahim *et al.* [126] by comparing it with column generation and OR-Tools on different instances. Later, Kool *et al.* [118] introduced a multi-attention mechanism to pass weighted information among different nodes out of consideration for the influence of adjacent structures on solutions. This attention mechanism enables nodes to capture more information from their own neighborhoods; hence, their encoder can learn more useful knowledge to obtain better solutions.

Both Nazari *et al.* [125] and Kool *et al.* [118] laid the research foundation for the follow-up development of the VRP. Therefore, many scholars either extended their models to solve VRP variants or modified them to obtain better solutions. Peng *et al.* [127] considered that node features should be constantly updated during the solving process. Therefore, they built a dynamic attention mechanism model (AM-D) by recoding embeddings at each step. Compared with Kool *et al.* [118], the performance of AM-D is notably improved for VRP20 (2.02%), VRP50 (2.01%) and VRP100 (2.55%). Based on the model of Nazari *et al.* [125], Duan *et al.* [128] added a classification decoder based on MLP to classify edges. The decoder of Nazari *et al.* [125] was used to output a sequence of nodes, while Duan *et al.* [128] used this sequence as labels to train another classification decoder for the final solutions. This type of joint training approach outperforms other existing LBO models in solving large-scale CVRP by approximately 5%. Vera and Abad [129] used one more encoder than Kool *et al.* [118] to embed vehicle information for the capacitated multi-vehicle routing problem (CMVRP) with a fixed fleet size. The VRPTW has also been widely studied, and the work of Falkner and Schmidt-Thieme [130] can be considered the first extension of Kool *et al.* [118] to solve the VRPTW. They employed two additional encoders to embed current tours and vehicles for solving large-scale CVRP and produce a comprehensive context for the decoder. Their decoder would concurrently select the visiting node and the serving vehicle. Their model exhibits strong results on solving VRPTWs, even outperforming OR-Tools.

In the past two years, there has been an increase in the publication of papers that design end-to-end models for multi-path problems. Similar to Peng *et al.* [127], in 2020, Xin *et al.* [131] proposed the dynamic update of embeddings before decoding. However, they considered the computational complexity of the model; they changed the attention weight of the visited nodes in the top layer of the encoder instead of re-embedding. In 2021, Xin *et al.* [132] proposed another approach to expand the model of [118] by using a multi-decoder with different parameters to train multiple policies and select the best one to decode at each step. Experiments indicated that these innovations are useful in improving the

original LBO model. Zhang *et al.* [133] used the model of Kool *et al.* [118] to solve multi-vehicle routing problems with soft time windows (MVRPSTW). They considered vehicles as multi-agents, where all agents share one encoder but use different decoders. Zhao *et al.* [134] modified the model proposed by Nazari *et al.* [125] by using a routing simulator to generate data and update both the dynamic state and mask for the CVRP and the VRPTW. Furthermore, they applied a local search to improve solutions of the LBO model, and they demonstrated that combining the LBO algorithms and heuristic search can be a general method of solving combinatorial problems. In contrast to previous LBO models, which are based on the standard policy gradient method, Sultana *et al.* [135] first proposed to add an entropy regularization term to encourage the exploration for solving VRPs, thereby avoiding the limitation that the LBO model can easily converge too rapidly to a poor solution. They experimentally demonstrated that the policy with the highest entropy was easier to find a satisfactory solution. Drori *et al.* [136] transformed the structure of the VRP into a line graph by defining each edge in the primal graph corresponding to a node in the line graph, and computed edge weights as node features. To effectively solve dynamic and stochastic VRP (DS-VRP), Bono *et al.* [137] proposed a multi-agent model by adding two attention networks based on [118] to encode vehicle's features and last decision. Their model presented favorable results for small-scale DS-CVRP and DS-CVRPTW compared with OR-Tools. However, the model exhibited poor generalization when testing on deterministic CVRPs and CVRPTWs. Lin *et al.* [138] incorporated the model of Nazari *et al.* [125] with a graph embedding layer to solve the electric vehicle routing problem with time windows (EVRPTW). They trained the model using the modified REINFORCE algorithm proposed by Kool *et al.* [118], but the maximum number of nodes in the test instance was only up to 100. Recently, Li *et al.* [139] modified the model proposed by Kool *et al.* [118] to solve the pickup and delivery problem (PDP), which has priority constraints and specific pairing relations. To learn complex relations and precedence among nodes of different roles, they added another six types of attention mechanisms based on the original attention mechanism. Later, aiming at that most end-to-end approaches are designed for homogeneous vehicle fleet, Li *et al.* [140] proposed to add a vehicle decoder to minimize the travel time among all heterogeneous vehicles based on the model of Kool *et al.* [118]. Their LBO model would select both a serving vehicle and a visiting node rather than solely selecting the next node to visit at each step.

Many scholars have proposed other novel end-to-end frameworks for VRPs with different objectives. Li *et al.* [141] and Yu *et al.* [142] proposed a model to plan online vehicle routes to minimize computation time. The differences between two papers lie in the architecture and problem background. Li *et al.* [141] first used an LSTM network to predict future traffic conditions and then used a double-reward value iterative network to make decisions. However, Yu *et al.* [142] planned routes by improving *Pointer Net* [85]. In addition, Li *et al.* [141] trained their model on data of 400,000 taxi trajectories in Beijing, whereas Yu *et al.* [142] applied to the green

logistics system and trained their model on the traffic data of Cologne, Germany. Balaji *et al.* [143] combined distributed prioritized experience replay [144] with DQN to maximize the total reward of the VRP. Although DQN is widely used in LBO models, it requires a significant amount of time to converge because the information update lag of the experience replay. With respect to this limitation, Mukhutdinov *et al.* [145] proposed using SL to generate preliminary Q values for nodes and they applied this approach to minimize the cost of the packet routing problem. Ramachandran Pillai *et al.* [146] designed an adaptive extended spiking neural P system with potentials (ATSNPS) to determine the shortest solutions for the VRPTW. They experimented on supermarket chain instances and demonstrated that their model can obtain better solutions. Sheng *et al.* [147] according to the principle of maximizing the total benefit, introduced global attention [148] to modify Pointer Net [85] to solve the VRP with task priority and limited resources (VRPTPLR). Cao *et al.* [149] first used the RL model to solve the stochastic shortest path (SSP) problem requiring on-time arrival. They used the Q-value to represent the probability of arriving on time and set the discount factor of reward as 1 to maximize the probability of arriving on time. Chen *et al.* [150] built an LBO model for an autonomous vehicle fleet and post-processed routes for minimizing the energy cost of the entire fleet. However, the environment of their model was too idealistic, and they did not consider the dynamics in the real world.

In addition to designing models for the VRP with a single objective, some studies aimed at achieving multiple objectives simultaneously, which is difficult for traditional algorithms. To minimize driving time and route length, Kalakanti *et al.* [151] proposed a framework with two stages, which included clustering by heuristics and route planning by Q-learning. However, experiments on three VRP variants demonstrated that their model performed poorly in a stochastic setting. To minimize the tour length and cost of the ride-sharing field, Holler *et al.* [152] used a MLP to compute the pooling weights and selected action based on the pooling mechanism. They used the DQN and PPO algorithm to train the LBO model respectively, and experiments demonstrated that DQN is more efficient.

C. Analysis of literatures

As we indicated previously, many papers that using the LBO algorithms to solve different VRP variants have been published (Fig. 4). There are 14 and 25 relevant papers were respectively published in 2019 and 2020. As we can observe, there has been a rapid expansion in this field over the last two years. This is primarily because of the following reasons: 1) an increasing number of scholars have proved that the LBO algorithm is competitive in solving combinatorial optimization problems; 2) with the development of economic globalization, transportation efficiency has become a key factor affecting company profits; 3) recent VRP studies have been characterized by large-scale online planning and complex constraints. These factors significantly promote the LBO algorithms in solving the VRP. Another aspect worth noting in Fig. 4 is that

scholars tended to use step-by-step approaches in the initial research phase of the field. However, they preferred end-to-end approaches after 2017, which is primarily owing to the history-making success achieved by Alpha GO. However, with the recent in-depth research in using the LBO algorithms for the VRP, many literatures have demonstrated that combining LBO model with other algorithms is more effective for complex optimization problems. Hence, the research on the step-by-step methods has been revived.

As shown in the pie chart on the right of Fig. 5, most of studies focus on the TSP (approximately 44%). As a widely studied variants of the VRP, the TSP is a basic graph problem; therefore, scholars tend to test a new model on the TSP and then extend the model to other VRP variants. The second widely studied variant is the CVRP (approximately 28%), which has a simple mathematical model but strong flexibility. Both the CVRP and TSP primarily target minimizing the tour length; hence, length is the most studied objective among the distribution in the objectives of the literatures (approximately 75.7%, Fig. 6). Note that the sum of percentages in Fig. 6 is greater than 1 because some studies are multi-objective.

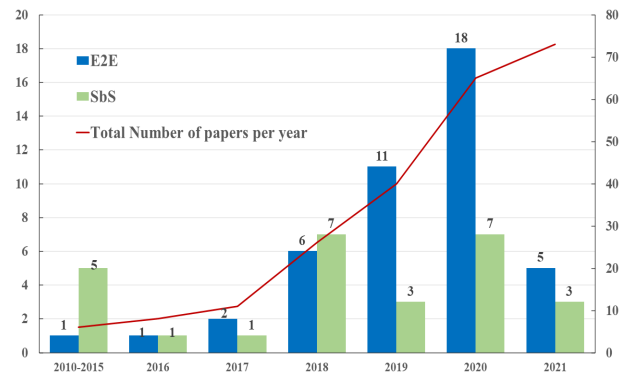


Fig. 4. Distribution of published papers per year for the VRP (the deadline for statistical data in 2021 is September).

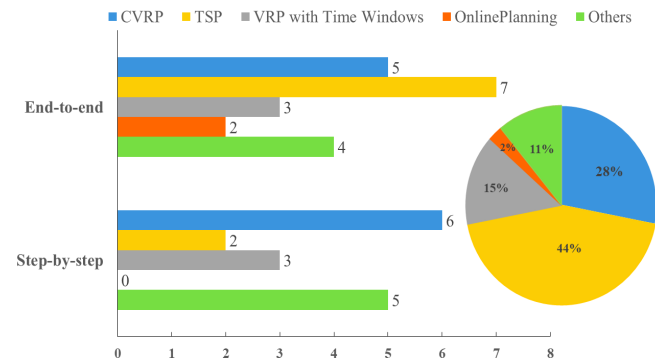


Fig. 5. Distribution of different variants.

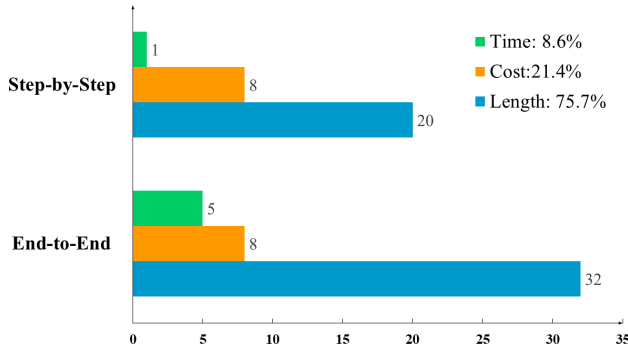


Fig. 6. Distribution of objective functions.

We summarize the characteristics of different VRP problems by synthesizing and analyzing the referenced papers. The four evaluation indicators are described as follows.

- **High-complexity.** High-complexity refers to VRPs with multiple constraints or objectives, and we quantifiably represent the practicability of an LBO model by complexity of its application problem because the VRP in the real world often has multiple constraints or objectives. In this paper, we consider that problems with two or more objectives or constraints are high complex. For example, the TSPTW with rejection solved by Zhang *et al.* [117] needs to minimize the tour length and the rejection rate; the VRP solved by Bono *et al.* [137] requires time windows and stochastic demands to be satisfied simultaneously. We considered that the above problems are complex, and we compared the number of studies of two class approaches applied to complex problems. We observed that step-by-step approaches are more likely to be selected for problems with high-complexity.
- **Stochastic.** We define the VRPs whose customer's demands, time windows or other uncertain elements follow some probability distribution models as stochastic problems. For example, Balaji *et al.* [143] considered a VRP variant of on-demand delivery, whose orders were generated with a constant probability; Joe and Lau [78] considered a real-time VRP in urban logistics in which customers and orders were randomly added or cancelled. Step-by-step approaches have limitations in solving stochastic VRPs since this type of the LBO algorithms needs customers and travel costs to be known in advance and longer solving time. Compared with step-by-step approaches, end-to-end approaches have the advantages of flexibility and time efficiency under stochastic environments, which can quickly response to changes by utilizing knowledge extracted from past training experience.
- **Timeliness.** Timeliness refers to those VRPs that require to plan a tour with the least time, which is a significant characteristic of real-world VRPs. We consider problems that use LBO approaches to minimize tour time are timeliness, such as in [142]. Overall, end-to-end approaches are easily used for problems with timeliness because of their quick solution speed.

- **Fuzzy.** With the development of the VRP, many new VRP variants have been proposed. We use fuzzy to define these new problems because researchers do not have much expertise in these problems. The more likely a type of the LBO approaches is used to solve new problems indicates that it has a more generic modeling framework. There are more fuzzy problems are solved by end-to-end approaches among our referred literatures since the learning process of these approaches do not require abundant domain knowledge.

Generally, end-to-end approaches are suitable for VRPs with stochastic or time requirements; step-by-step approaches can solve more complex problems effectively.

In addition to analyzing the applied problems of the LBO algorithms, it is interesting to compare their models. The previous review clearly indicates that many studies used the encoder-decoder framework; therefore, we compared these encoder-decoder models (see Table I, where MHA refers to the multi-head attention layers and FF refers to the feedforward network). We observed that the encoder-decoder framework is more commonly used in end-to-end approaches, and most differences of frameworks are in the encoder. This is probably because if scholars require different information, they must use different NNs to extract related features from the input. If authors seek to incorporate more features in addition to the node coordinates and demands, they often consider the GCN as a good option. Conversely, there are two main types of decoders: RNN with a pointing mechanism (PM) (Vinyals *et al.* [85]) and the other is composed of several multi-head attention sublayers (MHA) (Kool *et al.* [118]). In terms of learning manner, Vinyals *et al.* [85] used SL to train the model, whereas all the others used RL. This is expected because the VRP is an NP-hard problem, and it is difficult to obtain labels for training data. We also present an overview of the LBO architectures in Fig. 7. We first classify NN models according to training algorithms. Then, we list solving problems of different structures together with the corresponding literature and their publication time.

To clearly compare the literature, we list the main content of the referenced papers, including model features, baselines, and benchmarks in experiments at the end of the paper (see Table VII). For convenience, we abbreviate end-to-end approaches as E2E and step-by-step approaches as SbS in this table. Benchmarks can frequently be divided into simulation data and data from the literature or the real world; we refer to the latter collectively as real data.

IV. EXPERIMENTAL STUDY

A. Experimental Setting

Experimental algorithms. As described in Section III, the research of Lu *et al.* [82] (L2I) and Chen and Tian [81] (Rewriter) are the recent research trends of the step-by-step approaches, and both papers are also among the most cited step-by-step approaches. Kool *et al.* [118] (AM) made a significant contribution to solving the VRP by using an end-to-end framework. Many studies have modified the model of Kool *et al.* [118], and the model of Xin *et al.* [131] (ASWTAM),

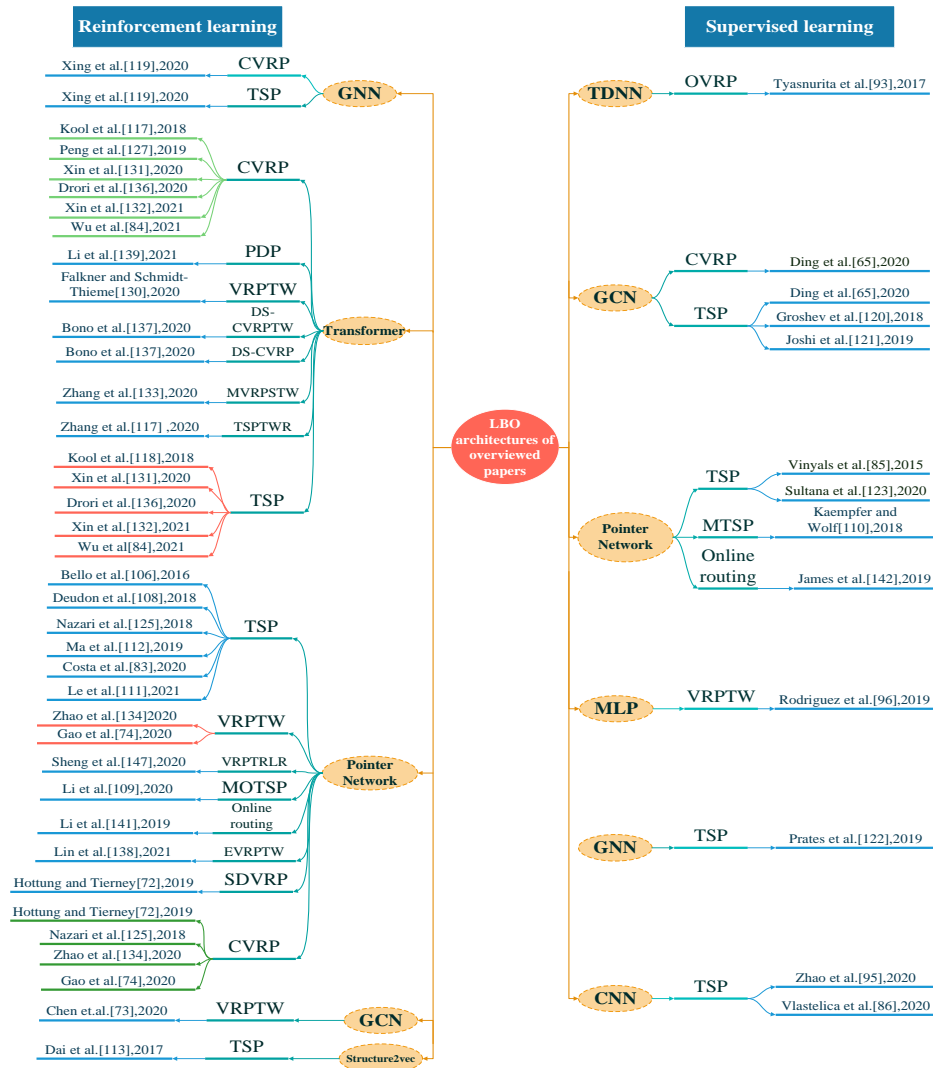


Fig. 7. Overview of LBO architecture of overviewed papers ([154]).

proposed in 2020, is one of the representatives. ASWTAM designs a step-wise update embedding mechanism, which is beneficial for the original AM model to focus on useful inputs and determine the optimal solution. To analyze the characteristics and limitations of different LBO models, we test these four representative LBO algorithms on different sizes of instances, and we compare them with other algorithms in this section. We select three classical meta-heuristics (ACO, TS and LNS), and two well-known solvers (Gurobi and OR-Tools) as baseline algorithms. The best parameters of different scales of the problems are determined through multiple experiments. To limit the total time expended when solving the problem, we modify the time limit parameter of Gurobi to 1800 seconds.

Problem Details. We evaluate four LBO models and baseline algorithms on the CVRP of different scales, and we do not distinguish the CVRP and VRP in this section.

Data generation.

(1)**Training set.** We consider three training instances, Euclidean VRP with 20, 50, and 100 nodes, named VRP20, VRP50, and VRP100, respectively. For all tasks, the location

of each customer and the depot are uniformly sampled in the unit square $[0,1]^2$, and the demand of each customer is also uniformly sampled from the discrete set $\{1, 2, \dots, 9\}$. The capacities of a vehicle are 30, 40, and 50 for $N = 20, 50$, and 100, respectively.

(2)**Test set.** We test different algorithms using three types of test data: 1) **Set 1**, following the same rules as the training set, we newly generate 300 instances for the three-scale VRP. 2) **Set 2** also contains 300 VRP instances with $n=20, 50$, and 100, but the locations of the nodes are sampled from the gamma distribution ($\alpha=1, \beta=0$). 3) **Set 3** contains nine benchmarks from Uchoa *et al.* [155], whose nodes ranged from 100 to 200.

Hyperparameters setting. The hyperparameters of the selected LBO models are the same as those in literatures to ensure the validity of the experimental results as much as possible. We set the random seed at 1234 to ensure the consistency of training data, and we set the batch size during testing to 1 to better compare the running time. Table II lists the other hyperparameters of the training model. We conduct

TABLE I
COMPARISON OF ENCODER-DECODER FRAMEWORKS AMONG REFERRED MODELS

Approaches	Literature	Problem	Encoder	Embedding features	Decoder	Learning manner
End-to-end Approaches	[128]	CVRP	GCN	nodes & distance matrix	RNN + PM	RL+SL
	[118]	CVRP	MHA	nodes	MHA	RL
	[85]	TSP	RNN	nodes	RNN + PM	SL
	[106]	TSP	RNN	nodes	RNN + PM	RL
	[107]	TSP	CNN	nodes	RNN	SL
	[109]	TSP	RNN	nodes	RNN + PM	RL
	[111]	TSP	GNN	nodes	RNN + PM	RL
	[112]	TSP	GNN	nodes	RNN + PM	RL
	[117]	TSPTWR	MHA	nodes	MHA	RL
	[123]	TSP	CNN	nodes	RNN + PM	SL
	[125]	CVRP	GCN	nodes	RNN + PM	RL
	[127]	CVRP	MHA	nodes	MHA	RL
	[129]	CMVRP	GCN	nodes	RNN + PM	RL
	[130]	CVRPTW	MHA	nodes & tour & vehicles	MHA	RL
	[131]	CVRP,TSP	MHA	nodes	MHA	RL
	[133]	MVRPSTW	MHA	nodes	MHA	RL
	[134]	CVRP,VRPTW	GCN	nodes	RNN + PM	RL
	[136]	CVRP,TSP	MHA	edges	MHA	RL
	[137]	DS-CVRP,DS-CVRPTW	MHA	nodes & vehicles	MHA	RL
	[138]	EVRPTW	GCN	nodes & battery & time & vehicles	RNN + PM	RL
[139]	PDP	MHA	nodes	MHA	RL	
[142]	Online VRP	GNN	nodes	RNN + PM	RL	
[147]	VRPTPLR	RNN	nodes	RNN + PM	RL	
[138]	EVRPTW	GCN	nodes	RNN + PM	RL	
Step-by-step Approaches	[72]	CVRP, SDVRP	Two linear layers	tour	FF	RL
	[74]	CVRP, CVRPTW	EGATE	nodes & edges	RNN + PM	RL
	[83]	TSP	GCN	tour	PM +Max-pooling & FF +Mean-pooling	RL

all the experiments on Python software on a computer with a Core i7-9800x 3.8-GHz CPU, 16 GB memory, Windows 10 operation system, and a single 2080Ti GPU.

TABLE II
THE HYPERPARAMETERS OF LBO MODELS

Parameters	AM	Rewriter	L2I	ASWTAM
The number of epochs	100	10	40000	100
The size of training set	1280000	100000	2000	1280000
Batch size	512	128	1000	512
Learning rate	0.001	5.00E-05	0.001	1.00E-04

Evaluation metrics. To better evaluate LBO models, we use multiple evaluation metrics that are widely used in the RL research community [25, 143]. In addition, since each type of VRP in set 1 and 2 only contains 300 instances, we use the Wilcoxon signed-rank test and Friedman test on test results to infer the holistic performance of different LBO models.

- Training time and occupy memory.
- Length of solutions, optimal gap, and solving time in per- instance.
- Rank obtained by Friedman test and p -value obtained from the Wilcoxon test.

Experimental design.

To fully evaluate the effect of LBO models, we compare the experimental algorithms from three aspects: time efficiency, scalability, and optimality. The comparison experiments can be divided into three parts:

Part I To validate the learning effectiveness of the LBO models, we train the LBO models on the training set and compare the training time, occupied memory, and learning curves among the LBO models.

Part II To validate the time efficiency and optimality of the LBO models, we test algorithms on set 1 and 2 and compare the solution length, solving time, and standard deviation. In addition, we use the Wilcoxon signed-rank test and Friedman test to analyze statistical results in depth.

Part III To validate the scalability of LBO models to larger problems, we test LBO models trained on VRP20 on set 3. We use the solution length, solving time, and optimal gap as evaluation indicators.

B. Results and Analysis

1) *Comparison and discussion of Part I:* We compared the training time (in hours) of the four LBO models on VRP20, VRP50, and VRP100. The left bar chart of Fig. 8 shows that end-to-end approaches frequently requires less training time than step-by-step approaches. AM requires the least training time, whereas Rewriter requires the most training time. This is primarily because these step-by-step approaches require to be combined with heuristic search to determine optimal solutions; therefore, they require more time to constantly experience the circulation of generation, evaluation, and evolution during training. While end-to-end approaches use NNs to replace this complex circulation, they can cost less time to learn. Comparing Rewriter with L2I, although Rewriter uses only 2-opt to generate neighborhood solutions and L2I uses six heuristics, the training size of Rewriter is approximately 50 times than that of L2I. Therefore, Rewriter requires more time than L2I. Comparing two end-to-end approaches, ASWTAM costs longer training time since it needs to step-wise update embeddings at each step.

We also recorded the occupied memory of the LBO models during training (right bar chart of Fig. 8), and we observed that step-by-step approaches require less memory than end-to-end approaches; e.g., L2I requires only approximately 1.2892 GB on VRP100 and ASWTAM requires 9.07 GB. This result reveals that end-to-end approaches can fully utilize advanced computing hardware to reduce training time but require the computer to have a large amount of memory to store sizable data generated by parallel computing. In addition, we observed that LBO models require more memory as the size of the problem increases. This is apparent because a large-scale VRP has high input dimensions, and solving it requires deeper NNs with more parameters. However, the problem size remains a challenge for the LBO algorithms to settle because of the curse of dimensionality and the limitations in computational resources. Comparing L2I with Rewriter, Rewriter requires more memory during training because it uses another policy network to select segments to be rewritten in addition to a rule-defining policy network. Similarly, we observed that ASWTAM requires more memory than AM. This is because ASWTAM requires step-wise update embeddings but the embeddings of AM are fixed.

We depicted the comparison of learning curves in Fig. 9, and we defined the average solution length of samples of a batch as distance. The learning curve is an important index for evaluating the learning ability of an LBO model, and it can provide much information about the LBO model. The earlier the turning point of the curve tends to be stable, the fewer samples are required for model training. In addition, the distance in the training process predicts the optimization of the model on the training data, whereas a lower value indicates a better performance of the LBO model. Comparing the two approaches, we can conclude that step-by-step approaches can converge faster than end-to-end approaches. Among them, L2I has the best learning performance because its learning curve reaches the turning point earlier, which proves that using heuristics as search operators can effectively improve the learning effectiveness of LBO models. This also explains why step-by-step approaches require less training data than end-to-end approaches. In addition, we can speculate that there is a mutual promotion between LBO models and heuristic algorithms, but exceedingly few heuristics may reduce the learning ability of LBO models, similar to Rewriter. Additionally, note that the learning curve of ASWTAM is below the AM as the training progressed. This proves that the input features have an important influence on the learning ability of the model because ASWTAM modifies AM by dynamically updating the embeddings.

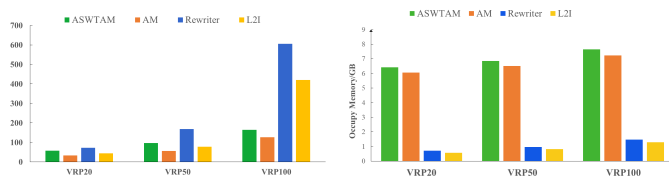


Fig. 8. Training time and occupy memory of LBO models on VRP20, VRP50, and VRP100.

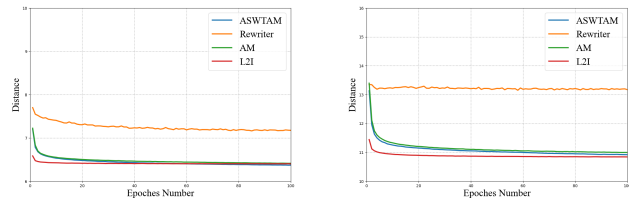


Fig. 9. Learning curves of LBO models on VRP20 and VRP50.

2) *Comparison and discussion of Part II:* Table IV and V show the optimization of the LBO algorithms on test sets 1 and 2, respectively. For the columns in Table IV and V, column 1 shows the algorithms, and columns 2-4 respectively list the average tour length (mean), standard deviation (std), and average solving time (time) used by each algorithm for instances with $n=20$. Columns 5-7 and 8-10 respectively list the same information of experimental algorithms on the instances with $n = 50$ and 100 . Note that we tested two types of search policy for AM for a comprehensive comparison.

First, the tables show that L2I has the minimum mean on testing instances except for VRP20, and its results are even better than those of OR-Tools. Furthermore, we observed that the standard deviations of L2I and Rewriter are always smaller than those of OR-Tools, LNS and ACO. These results indicate that incorporating the LBO algorithms within the heuristic search can result in a stable performance and strong generalization of the models. Several studies [108, 123, 134] have indicated this positive effect. Second, we observed that TS and LNS perform below ASWTAM and AM (greedy) in VRP50 and VRP100 on set 1, and ASWTAM is better than OR-Tools in VRP100. However, end-to-end approaches have disappointed performance on set 2, although their solutions are still better than that of ACO. Therefore, we can conclude that end-to-end approaches have a strong dependence on the distribution of data. In addition, comparing L2I to end-to-end approaches, although L2I outperforms ASWTAM and AM in terms of the quality of the solution, end-to-end approaches have an advantage in computational time, e.g., AM (greedy) requires only **0.93 s** and ASWTAM requires **1.82 s** on VRP100 from set 1, but L2I requires 25.25 s. Third, we observed that different search methods are applicable to different types of test data. Comparing Table IV and V, we can conclude that the sampling search method outperforms the greedy search method on data obeying the gamma distribution, but the greedy search method can obtain better solutions for data obeying the same distribution as the training data. This is because the greedy search selects the best action at each step according to the training experience, whereas the sampling search selects the best from many sampled solutions [112]. Hence, a greedy search is more dependent on the data distribution. Finally, comparing L2I with Rewriter, we observed that L2I has better optimization than Rewriter. Thus, more search operators are beneficial to searching for a larger solution space, and the optimal solution is more likely to be determined. However, note that an excessive number of heuristic operators result in the solving time of L2I being twice as that of Rewriter. We also compared two LBO models of end-to-end approaches and

concluded that ASWTAM is better than AM in quality of solutions. ASWTAM adopts a dynamic embedding mechanism, and dynamic embedding aids the network in capturing the real-time characteristics of the environment.

To further illustrate the overall performance of the experimental algorithms across all test cases, we performed nonparametric tests on VRP20 from set 1 and 2, and the results are shown in Fig. 10 and Table III. Fig. 10 shows the results of the Friedman test and uses Friedman Rank as the abscissa. The smaller the rank value is, the better the algorithm performs in all instances. Table III shows the results of the Wilcoxon test and column 1 represents the tested algorithms. The Wilcoxon test first computes the difference between the two algorithms on a set of instances and then obtains the corresponding rank by sorting the absolute values of the differences. Column 2 of Table III summarizes the rank of the positive differences, and column 3 summarizes the rank of negative differences. The gap between the two sums represents the difference between the two algorithms, and the p -value in the final column is used to evaluate this difference quantitatively. Under the confidence degree $\alpha=0.05$, a p -value greater than 0.05 means that there is no significant difference between the two algorithms.

From the Friedman test (in Fig. 10) on VRP20, we observed that L2I is second only to Gurobi in VRP20, and the results of the Wilcoxon test of the two algorithms in Table III indicates that L2I is not significantly different from Gurobi in VRP20 at a confidence coefficient of 0.05. Moreover, we observed that the performance of ASWTAM is not significantly different from AM on set 1, but AM is distinctly inferior to ASWTAM on dataset 2. This proves that AM is effectively improved by ASWTAM. It is worthy noting that the p -value of OR-Tools and L2I is 1 in set 2, although this value is 0.000002 in set 1. Similar results are observed in the comparison of the other three LBO models with OR-Tools, in which LBO models achieve p -values >0.05 in set 1 but <0.05 in set 2. This indicates that LBO models have limitations when applied to the problem whose data distribution is different from the training set.

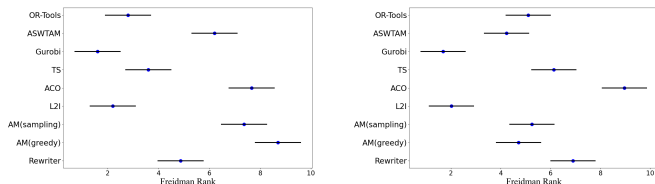


Fig. 10. Friedman test on VRP20 from different datasets (left from set 1, right from set 2).

3) *Comparison and discussion of Part III:* We trained LBO models on VRP20 and tested them on set 3 to verify the scalability of the LBO models, and we depicted the performance of the experimental algorithms in Table VI and Fig. 11. We selected only AM with a sampling search policy for testing.

Note that the solving time of all the algorithms increase as the scale of the problem increased, but the solving time of LBO is always significantly less than that of ACO and TS. The most time-consuming L2I is 33.08 s in instances

TABLE III
WILCOXON TEST RESULTS AMONG DIFFERENT ALGORITHMS FOR $\alpha=0.05$
ON VRP20 FROM DIFFERENT DATE SET.

Comparing Algorithms	Data Set 1			Data Set 2		
	R+	R-	p -value	R+	R-	p -value
L2I vs OR-Tools	465	0	0.000002	267	198	0.471592
L2I vs Gurobi	168	267	1.000000	82.5	382.5	1.000000
L2I vs TS	465	0	0.000002	465	0	0.000002
Rewriter vs ACO	465	0	0.000002	465	0	0.000002
Rewriter vs TS	109	356	1.000000	3	462	0.000002
Rewriter vs OR-Tools	181	284	1.000000	53	412	0.000053
Rewriter vs AM(sampling)	29	436	1.000000	465	0	0.000002
Rewriter vs AM(greedy)	24	441	1.000000	465	0	0.000002
ASWTAM vs Rewriter	434	31	0.000033	3	462	0.000002
ASWTAM vs ACO	465	0	0.000002	452	13	0.000006
ASWTAM vs AM(greedy)	285	150	0.130731	464	1	0.000002
ASWTAM vs OR-Tools	316	149	0.084035	0	465	0.000002
AM(sampling) vs AM(greedy)	219	246	1.000000	443	22	0.000014
AM(greedy) vs ACO	465	0	0.000002	15	450	0.000012
AM(greedy) vs TS	393	72	0.000928	0	465	0.000002
AM(greedy) vs OR-Tools	313	152	0.095706	0	465	0.000002

with 195 nodes, whereas TS requires 319.98 s; the longest computational time of the end-to-end model is **less than 7 s**, which is significantly less than that of OR-Tools. Second, L2I has the best scalability on benchmarks, and the gap between L2I and the optimal does **not exceed 0.1**. However, L2I is still inferior to TS in X-n101-k25 and X-n186-k15. Moreover, as shown in Table VI and Fig. 11, models of end-to-end approaches are more unstable than the step-by-step approaches for large-scale problems. The maximum gap of AM is up to 5.99, but the minimum gap is only 0.83. However, compared with AM, the ASWTAM exhibits significant improvement, and it is better than Rewriter in X-n186-k15. In the other instances, the solutions of ASWTAM are always better than ACO and close to Rewriter. This indicates that end-to-end approaches are promising, and further research is required to achieve better improvement.

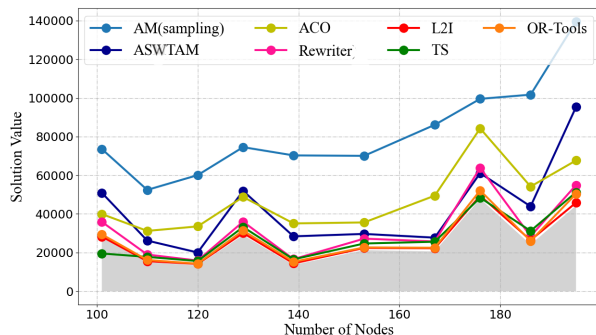


Fig. 11. Comparison of solution value of different algorithms on dataset 3.

4) *Experiments conclusion:* To demonstrate the effectiveness of the LBO models, we tested AM, ASWTAM, Rewriter, and L2I on three datasets and compared the LBO approaches with ACO, TS, LNS, Gurobi, and OR-Tools. According to the results of the three experiments, the following conclusions can be drawn: (a) step-by-step approaches have faster convergence and better generalization than end-to-end approaches, but they require more computational time; (b) end-to-end approaches frequently have less computational time both during training and testing, but they require more computational resources; (c) both approaches have limitations in the scale of problems and their performance is affected by data distribution; (d) compared with conventional algorithms, LBO approaches require

TABLE IV
COMPARISON OF AVERAGE VALUE AND SOLVING TIME (IN SECONDS) OF DIFFERENT ALGORITHMS ON DATASET 1.

Baseline	VRP20, Cap30			VRP50, Cap40			VRP100, Cap50		
	mean	std	time	mean	std	time	mean	std	time
Gurobi	5.74	0.62	1800		-			-	
OR-Tools	6.12	1.06	1.25	10.55	1.56	2.32	16.55	2.06	3.12
TS	6.27	0.69	23.57	11.4	0.96	54.53	18.61	1.86	113.4
ACO	12.29	1.38	7.42	19.86	1.97	34.81	37.03	4.57	126.6
LNS	6.48	0.96	4.96	12.85	1.62	94.32	18.86	2.03	771.6
AM(greedy)	6.41	0.83	0.25	10.79	0.82	0.53	16.66	1.78	0.93
AM(sampling)	6.42	0.83	0.3	10.9	0.92	0.53	19.35	1.65	0.91
ASWTAM	6.36	0.67	1.41	11.03	0.76	1.65	16.36	1.58	1.82
Rewriter	6.83	0.92	2.15	12.45	1.03	5.14	19.98	1.9	10.56
L2I	6.1	0.73	5.93	10.34	0.86	13.33	16.11	1.8	25.25

TABLE V
COMPARING AVERAGE VALUE AND SOLVING TIME (IN SECONDS) OF DIFFERENT ALGORITHMS ON DATA SET 2.

Baseline	VRP20, Cap30			VRP50, Cap40			VRP100, Cap50		
	mean	std	time	mean	std	time	mean	std	time
Gurobi	20.85	5.47	1800		-			-	
OR-Tools	21.83	5.58	1.03	38.16	9.01	2.12	58.28	12.31	2.56
TS	22.51	5.73	58.86	38.47	9.03	76.26	63.85	12.38	113.93
ACO	35.13	8.61	23.51	87.03	13.99	45.15	165.79	13.95	79.08
LNS	21.09	5.80	20.15	41.85	10.56	103.84	62.12	13.51	804.10
AM(greedy)	40.33	13.9	0.42	149.17	73.03	1.03	241.25	138.26	0.98
AM(sampling)	37.83	9.54	0.35	95.23	21.27	0.62	169.13	30.31	0.8
ASWTAM	28.07	7.54	1.41	58.02	16.49	1.94	110.79	22.85	2.2
Rewriter	24.45	6.63	2.26	44.05	9.95	5.83	69.46	12.71	12.93
L2I	20.88	5.41	6.01	35.43	8.75	13.6	54.44	12.35	25.13

further improvements.

V. CONCLUSION AND RESEARCH TREND

The LBO algorithms have been successfully applied to a series of optimization problems, and studies on using these algorithms to solve the VRP can be divided into two types: end-to-end and step-by-step approaches. Exhaustive experiments demonstrate that step-by-step approaches have strong scalability but are not suitable for problems requiring solving time, whereas end-to-end approaches can rapidly solve problems but are more dependent on data distribution.

Using the LBO algorithms to solve the VRP is still under research. There are several challenges in the LBO algorithms need to be settled in the future and we suggest several potential research directions of applying the LBO algorithms in the VRP from these limitations.

- 1) **Solving sizable or more complex VRPs based on the decomposition framework.** In 2014, Yao and Liu [156] had proposed that scaling up learning algorithms is an important problem. LBO models have difficulty in solving combinatorial optimization problems with high complexity or large scale, because these problems result in a curse of dimensionality and easy to overfit [157]. Li *et al.* [109] proposed to decompose a multi-objective VRP into a set of subproblems, and their results demonstrated

that their framework is better than NSGA-II in problems with five objectives; Fu *et al.* [116] also decomposed a sizable TSP into multiple small-scale subproblems and solved each of them using the SL model, and they can effectively solve the TSP with 10000 nodes. Hence, decomposing complex or sizable problems to multiple simple subproblems and solving them seems to be a feasible approach.

- 2) **Combining with conventional algorithms to improve the generality of LBO models.** Nickel *et al.* [158] illustrated that the relation learned by an LBO model in knowledge graphs only makes sense when applied to entities of the right type. We also observed in our experiments, LBO models are affected by data distribution, while combining heuristics as search operators can improve the generality of the LBO algorithms. Hence, considering different forms of combination might be beneficial to guaranteeing a better generalization of LBO models. Using a solver to post-process solutions of LBO models [86] or using solutions generated by heuristic algorithms to pretrain LBO models [120] are all good research directions.
- 3) **Embedding other algorithms to increase training efficiency of LBO models.** Butler *et al.* [159] indicated that the data limitations of the LBO algorithms must be

solved. The LBO algorithms frequently require sufficient data as a training set, but it is difficult to obtain sufficient raw data of combinatorial optimization problems in real world. What's more, the value function of learning models is randomly initialized, and models would select a random action with a certain probability to balance exploration and exploitation. LBO models might require a large amount of time and data to train but still converge to a suboptimal solution. Hence, embedding other algorithms to accelerate convergence of LBO model is necessary. For example, using other algorithms computes the initial value function [126, 145] or generates the prior knowledge for the next action [119].

- 4) **Building a generic framework for the LBO algorithms to solve VRPs.** Cappart *et al.* [160] proposed to build a generic modeling framework for LBO algorithms as a new direction; Wagstaff [161] also indicated that the LBO algorithms have not yet be such matured that researchers from other areas can simply apply them. Although many LBO algorithms have been proposed to solve different VRPs, encapsulating different LBO algorithms as a solver is still difficult for researchers. Many technical difficulties require to be solved and the most critical step is to define a generic modeling framework to provide upper-application with uniform interface.

REFERENCES

- [1] T. Pinto, C. Alves, and J. V. de Carvalho, "Models and advanced optimization algorithms for the integrated management of logistics operations," in *Congress of APDIO, the Portuguese Operational Research Society*. Springer, 2017, pp. 313–324.
- [2] R. S. Khan, J.-B. Yang, and J. Handl, "Interactive multi-objective vehicle routing via ga-based dynamic programming," in *2015 International Conference on Transportation Information and Safety (ICTIS)*. IEEE, 2015, pp. 318–322.
- [3] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 1, pp. 70–85, 2016.
- [4] P. Toth and D. Vigo, "An overview of vehicle routing problems," *The vehicle routing problem*, pp. 1–26, 2002.
- [5] J.-F. Cordeau, G. Laporte, M. W. Savelsbergh, and D. Vigo, "Vehicle routing," *Handbooks in operations research and management science*, vol. 14, pp. 367–428, 2007.
- [6] D. Pisinger and S. Ropke, "A general heuristic for vehicle routing problems," *Computers & operations research*, vol. 34, no. 8, pp. 2403–2435, 2007.
- [7] J. Euchi and H. Chabchoub, "A hybrid tabu search to solve the heterogeneous fixed fleet vehicle routing problem," *Logistics Research*, vol. 2, no. 1, pp. 3–11, 2010.
- [8] M. Guzairov, N. Yusupova, O. Smetanina, and E. Y. Rassadnikova, "Models and algorithms for the vehicle routing problem with time windows and other conditions," in *2016 13th International Scientific-Technical*

TABLE VI
THE PERFORMANCE OF DIFFERENT ALGORITHMS ON DATA SET 3. THE GAP IS BETWEEN EXPERIMENTAL ALGORITHMS AND THE OPTIMAL SOLUTION GIVEN BY THE INSTANCE. THE SOLVING TIME IS COMPUTED IN SECONDS.

Instance	Optimal	ASWTAM		Rewriter		AM		L2I		ACO		TS		OR-Tools							
		Value	Gap	Value	Gap	Value	Gap	Value	Time	Gap	Value	Time	Value	Time	Value	Gap					
X-n101-k25	27591	50934	4.81	35879	11.24	0.30	73578	3.95	0.83	28247	24.23	0.02	39913	61.21	0.5	19504	188.89	-0.29	29405	0.53	0.06
X-n110-k13	14971	26176	4.83	18948	11.13	0.26	52410	3.45	2.91	15526	24.1	0.02	31192	50.03	1.1	17745	211	0.18	16149	3.43	0.08
X-n120-k6	13332	20020	4.32	15848	10.97	0.23	60056	3.89	5.99	14095	25.53	0.05	33517	57.26	1.5	15591	208.53	0.17	14243	4.76	0.07
X-n129-k18	28940	51904	4.68	35966	12.49	0.17	74536	4.34	2.55	30093	30.19	0.05	48745	51.47	0.7	33196	309.08	0.15	31362	10.02	0.08
X-n139-k10	13590	28382	5.48	16579	12.54	0.23	70266	4.31	4.39	14394	28.06	0.04	35071	70.86	1.6	16479	315	0.21	15223	11.28	0.12
X-n153-k22	21220	29631	5.15	27183	14.61	0.28	70046	4.12	2.09	22364	24.06	0.06	35577	94.01	0.7	24678	444.8	0.16	22650	23.4	0.07
X-n176-k26	20557	27697	5.26	25716	15.08	0.25	86097	4.69	3.18	22225	40.31	0.08	49431	85.54	1.4	25582	348.16	0.24	22477	22.85	0.09
X-n186-k15	47812	61107	6.1	63848	17.47	0.33	99533	5.53	1.08	49412	31.93	0.03	84249	142.9	0.8	48386	341.64	0.01	52111	44.38	0.09
X-n195-k51	24145	43872	6.19	28430	16.96	0.18	101705	6	3.21	26402	33.08	0.09	54150	136.3	1.2	31176	319.98	0.29	26017	54.37	0.07

- Conference on Actual Problems of Electronics Instrument Engineering (APEIE)*, vol. 2. IEEE, 2016, pp. 412–416.
- [9] H. Afaq and S. Saini, “A novel approach to solve graph based travelling salesman problem using particle swarm optimization technique,” in *2012 IEEE International Conference on Computational Intelligence and Computing Research*. IEEE, 2012, pp. 1–4.
- [10] M. A. Mohammed, M. S. Ahmad, and S. A. Mostafa, “Using genetic algorithm in implementing capacitated vehicle routing problem,” in *2012 International conference on computer & information science (ICIS)*, vol. 1. IEEE, 2012, pp. 257–262.
- [11] J. H. Wilck IV and T. M. Cavalier, “A construction heuristic for the split delivery vehicle routing problem,” 2012.
- [12] U. Ritzinger, J. Puchinger, and R. F. Hartl, “A survey on dynamic and stochastic vehicle routing problems,” *International Journal of Production Research*, vol. 54, no. 1, pp. 215–231, 2016.
- [13] T. Jastrzab and A. Buchcik, “Practical applications of smart delivery systems: Mine evacuation as an example of rich vehicle routing problem,” *Smart Delivery Systems*, pp. 249–268, 2020.
- [14] J. K. Lenstra and A. R. Kan, “Complexity of vehicle routing and scheduling problems,” *Networks*, vol. 11, no. 2, pp. 221–227, 1981.
- [15] B. L. Golden, S. Raghavan, and E. A. Wasil, *The vehicle routing problem: latest advances and new challenges*. Springer Science & Business Media, 2008, vol. 43.
- [16] J. Mańdziuk, “New shades of the vehicle routing problem: emerging problem formulations and computational intelligence solution methods,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 3, no. 3, pp. 230–244, 2018.
- [17] A. O. Adewumi and O. J. Adeleke, “A survey of recent advances in vehicle routing problems,” *International Journal of System Assurance Engineering and Management*, vol. 9, no. 1, pp. 155–172, 2018.
- [18] A. Dixit, A. Mishra, and A. Shukla, “Vehicle routing problem with time windows using meta-heuristic algorithms: a survey,” in *Harmony search and nature inspired optimization algorithms*. Springer, 2019, pp. 539–546.
- [19] W. Cao and W. Yang, “A survey of vehicle routing problem,” in *MATEC Web of Conferences*, vol. 100. EDP Sciences, 2017, p. 01006.
- [20] R. Goel and R. Maini, “Vehicle routing problem and its solution methodologies: a survey,” *International Journal of Logistics Systems and Management*, vol. 28, no. 4, pp. 419–435, 2017.
- [21] X.-D. Zhang, “Machine learning,” in *A Matrix Algebra Approach to Artificial Intelligence*. Springer, 2020, pp. 223–440.
- [22] I. El Naqa and M. J. Murphy, “What is machine learning?” in *machine learning in radiation oncology*. Springer, 2015, pp. 3–11.
- [23] K. A. Smith, “Neural networks for combinatorial optimization: a review of more than a decade of research,” *INFORMS Journal on Computing*, vol. 11, no. 1, pp. 15–34, 1999.
- [24] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, “State-of-the-art in artificial neural network applications: A survey,” *Heliyon*, vol. 4, no. 11, p. e00938, 2018.
- [25] A. François, Q. Cappart, and L.-M. Rousseau, “How to evaluate machine learning approaches for combinatorial optimization: Application to the travelling salesman problem,” *arXiv preprint arXiv:1909.13121*, 2019.
- [26] R. Bai, X. Chen, Z.-L. Chen, T. Cui, S. Gong, W. He, X. Jiang, H. Jin, J. Jin, G. Kendall *et al.*, “Analytics and machine learning in vehicle routing research,” *arXiv preprint arXiv:2102.10012*, 2021.
- [27] G. B. Dantzig and J. H. Ramser, “The truck dispatching problem,” *Management science*, vol. 6, no. 1, pp. 80–91, 1959.
- [28] G. Clarke and J. W. Wright, “Scheduling of vehicles from a central depot to a number of delivery points,” *Operations research*, vol. 12, no. 4, pp. 568–581, 1964.
- [29] G. Laporte, “What you should know about the vehicle routing problem,” *Naval Research Logistics (NRL)*, vol. 54, no. 8, pp. 811–819, 2007.
- [30] Laporte, Gilbert, “Fifty years of vehicle routing,” *Transportation science*, vol. 43, no. 4, pp. 408–416, 2009.
- [31] R. Fukasawa, H. Longo, J. Lysgaard, M. P. De Aragão, M. Reis, E. Uchoa, and R. F. Werneck, “Robust branch-and-cut-and-price for the capacitated vehicle routing problem,” *Mathematical programming*, vol. 106, no. 3, pp. 491–511, 2006.
- [32] M. Jünger, G. Reinelt, and G. Rinaldi, “The traveling salesman problem,” *Handbooks in operations research and management science*, vol. 7, pp. 225–330, 1995.
- [33] P. Toth and D. Vigo, “Models, relaxations and exact approaches for the capacitated vehicle routing problem,” *Discrete Applied Mathematics*, vol. 123, no. 1-3, pp. 487–512, 2002.
- [34] M. Fisher, “Vehicle routing handbooks on operations research and management science,” *Network Routing*, pp. 1–33, 1995.
- [35] J. Caceres-Cruz, P. Arias, D. Guimarans, D. Riera, and A. A. Juan, “Rich vehicle routing problem: Survey,” *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, pp. 1–28, 2014.
- [36] G. Laporte and Y. Nobert, “Exact algorithms for the vehicle routing problem,” in *North-Holland Mathematics Studies*. Elsevier, 1987, vol. 132, pp. 147–184.
- [37] G. Laporte, H. Mercure, and Y. Nobert, “An exact algorithm for the asymmetrical capacitated vehicle routing problem,” *Networks*, vol. 16, no. 1, pp. 33–46, 1986.
- [38] S. Eilon, C. D. T. Watson-Gandy, N. Christofides, and R. de Neufville, “Distribution management-mathematical modelling and practical analysis,” *IEEE Transactions on Systems, Man, and Cybernetics*, no. 6, pp. 589–589, 1974.
- [39] M. Rao and S. Zionts, “Allocation of transportation units to alternative trips—a column generation scheme

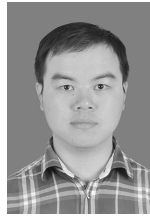
- with out-of-kilter subproblems,” *Operations Research*, vol. 16, no. 1, pp. 52–63, 1968.
- [40] N. Christofides and S. Eilon, “An algorithm for the vehicle-dispatching problem,” *Journal of the Operational Research Society*, vol. 20, no. 3, pp. 309–318, 1969.
- [41] N. Christofides, A. Mingozzi, and P. Toth, “Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations,” *Mathematical programming*, vol. 20, no. 1, pp. 255–282, 1981.
- [42] B. L. Golden, E. A. Wasil, J. P. Kelly, and I.-M. Chao, “The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results,” in *Fleet management and logistics*. Springer, 1998, pp. 33–56.
- [43] S. N. Kumar and R. Panneerselvam, “A survey on the vehicle routing problem and its variants,” 2012.
- [44] J.-F. Cordeau and G. Laporte, “Tabu search heuristics for the vehicle routing problem,” *Metaheuristic Optimization via Memory and Evolution*, pp. 145–163, 2005.
- [45] G. Kindervater, “Vehicle routing: handling edge exchange,” *Local Search in Combinatorial Optimization*, 1997.
- [46] S. Desale, A. Rasool, S. Andhale, and P. Rane, “Heuristic and meta-heuristic algorithms and their relevance to the real world: a survey,” *Int. J. Comput. Eng. Res. Trends*, vol. 351, no. 5, pp. 2349–7084, 2015.
- [47] G. Laporte and F. Semet, “Classical heuristics for the capacitated vrp,” in *The vehicle routing problem*. SIAM, 2002, pp. 109–128.
- [48] T. O. Ayodele, “Types of machine learning algorithms,” *New advances in machine learning*, vol. 3, pp. 19–48, 2010.
- [49] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.
- [50] L. P. Kaelbling, M. L. Littman, and A. W. Moore, “Reinforcement learning: A survey,” *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.
- [51] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [52] J. Yan, S. Yang, and E. R. Hancock, “Learning for graph matching and related combinatorial optimization problems,” in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*. International Joint Conferences on Artificial Intelligence Organization, 2020, pp. 4988–4996.
- [53] B. Mahesh, “Machine learning algorithms-a review,” *International Journal of Science and Research (IJSR).[Internet]*, vol. 9, pp. 381–386, 2020.
- [54] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, “The arcade learning environment: An evaluation platform for general agents,” *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, 2013.
- [55] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, “Mastering the game of go with deep neural networks and tree search,” *nature*, vol. 529, no. 7587, pp. 484–489, 2016.
- [56] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.
- [57] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in *International conference on machine learning*. PMLR, 2015, pp. 2048–2057.
- [58] M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A. M. Karimi-Mamaghan, and E.-G. Talbi, “Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art,” *European Journal of Operational Research*, 2021.
- [59] H. Song, I. Triguero, and E. Özcan, “A review on the self and dual interactions between machine learning and optimisation,” *Progress in Artificial Intelligence*, vol. 8, no. 2, pp. 143–165, 2019.
- [60] L. M. Gambardella and M. Dorigo, “Ant-q: A reinforcement learning approach to the traveling salesman problem,” in *Machine learning proceedings 1995*. Elsevier, 1995, pp. 252–260.
- [61] F. C. de Lima Junior, J. D. de Melo, and A. D. D. Neto, “Using q-learning algorithm for initialization of the grasp metaheuristic and genetic algorithm,” in *2007 International Joint Conference on Neural Networks*. IEEE, 2007, pp. 1243–1248.
- [62] M. M. Alipour, S. N. Razavi, M. R. F. Derakhshi, and M. A. Balafar, “A hybrid algorithm using a genetic algorithm and multiagent reinforcement learning heuristic to solve the traveling salesman problem,” *Neural Computing and Applications*, vol. 30, no. 9, pp. 2935–2951, 2018.
- [63] F. Liu and G. Zeng, “Study of genetic algorithm with reinforcement learning to solve the tsp,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 6995–7001, 2009.
- [64] T. Phiboonbanakit, T. Horanont, T. Supnithi, and V.-N. Huynh, “Knowledge-based learning for solving vehicle routing problem,” in *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, 2018, pp. 1103–1111.
- [65] J.-Y. Ding, C. Zhang, L. Shen, S. Li, B. Wang, Y. Xu, and L. Song, “Accelerating primal solution findings for mixed integer programs based on solution prediction,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 02, 2020, pp. 1452–1459.
- [66] Y. Sun, A. Ernst, X. Li, and J. Weiner, “Generalization of machine learning for problem reduction: a case study on travelling salesman problems,” *OR Spectrum*, vol. 43, no. 3, pp. 607–633, 2021.
- [67] P. Cooray and T. D. Rupasinghe, “Machine learning-based parameter tuned genetic algorithm for energy minimizing vehicle routing problem,” *Journal of Industrial Engineering*, vol. 2017, 2017.
- [68] F. Al-Duoli, G. Rabadi, M. Seck, and H. A. Handley, “Hybridizing meta-raps with machine learning algorithms,” in *2018 IEEE Technology and Engineering*

- Management Conference (TEMSCON)*. IEEE, 2018, pp. 1–6.
- [69] P. Shaw, “A new local search algorithm providing high quality solutions to vehicle routing problems,” *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*, vol. 46, 1997.
- [70] G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, and G. Dueck, “Record breaking optimization results using the ruin and recreate principle,” *Journal of Computational Physics*, vol. 159, no. 2, pp. 139–171, 2000.
- [71] D. Pisinger and S. Ropke, “Large neighborhood search,” in *Handbook of metaheuristics*. Springer, 2010, pp. 399–419.
- [72] A. Hottung and K. Tierney, “Neural large neighborhood search for the capacitated vehicle routing problem,” *arXiv preprint arXiv:1911.09539*, 2019.
- [73] M. Chen, L. Gao, Q. Chen, and Z. Liu, “Dynamic partial removal: A neural network heuristic for large neighborhood search,” *arXiv preprint arXiv:2005.09330*, 2020.
- [74] L. Gao, M. Chen, Q. Chen, G. Luo, N. Zhu, and Z. Liu, “Learn to design the heuristics for vehicle routing problem,” *arXiv preprint arXiv:2002.08539*, 2020.
- [75] M. M. Solomon, “Algorithms for the vehicle routing and scheduling problems with time window constraints,” *Operations research*, vol. 35, no. 2, pp. 254–265, 1987.
- [76] R. A. Bianchi, C. H. Ribeiro, and A. H. Costa, “On the relation between ant colony optimization and heuristically accelerated reinforcement learning,” in *1st international workshop on hybrid control of autonomous system*. Citeseer, 2009, pp. 49–55.
- [77] F. Yang, T. Jin, T.-Y. Liu, X. Sun, and J. Zhang, “Boosting dynamic programming with neural networks for solving np-hard problems,” in *Asian Conference on Machine Learning*. PMLR, 2018, pp. 726–739.
- [78] W. Joe and H. C. Lau, “Deep reinforcement learning approach to solve dynamic vehicle routing problem with stochastic customers,” in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 30, 2020, pp. 394–402.
- [79] A. Delarue, R. Anderson, and C. Tjandraatmadja, “Reinforcement learning with combinatorial actions: An application to vehicle routing,” *arXiv preprint arXiv:2010.12001*, 2020.
- [80] Y. Yao, Z. Peng, and B. Xiao, “Parallel hyper-heuristic algorithm for multi-objective route planning in a smart city,” *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10 307–10 318, 2018.
- [81] X. Chen and Y. Tian, “Learning to perform local rewriting for combinatorial optimization,” 2019.
- [82] H. Lu, X. Zhang, and S. Yang, “A learning-based iterative method for solving vehicle routing problems,” in *International Conference on Learning Representations*, 2019.
- [83] P. R. d. O. da Costa, J. Rhuggenaath, Y. Zhang, and A. Akcay, “Learning 2-opt heuristics for the traveling salesman problem via deep reinforcement learning,” *arXiv preprint arXiv:2004.01608*, 2020.
- [84] Y. Wu, W. Song, Z. Cao, J. Zhang, and A. Lim, “Learning improvement heuristics for solving routing problems..” *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
- [85] O. Vinyals, M. Fortunato, and N. Jaitly, “Pointer networks,” *arXiv preprint arXiv:1506.03134*, 2015.
- [86] M. Vlastelica, A. Paulus, V. Musil, G. Martius, and M. Rolínek, “Differentiation of blackbox combinatorial solvers,” *arXiv preprint arXiv:1912.02175*, 2019.
- [87] Y. Ma, J. Li, Z. Cao, W. Song, L. Zhang, Z. Chen, and J. Tang, “Learning to iteratively solve routing problems with dual-aspect collaborative transformer,” *Advances in Neural Information Processing Systems*, vol. 34, 2021.
- [88] W. Qin, Z. Zhuang, Z. Huang, and H. Huang, “A novel reinforcement learning-based hyper-heuristic for heterogeneous vehicle routing problem,” *Computers & Industrial Engineering*, vol. 156, p. 107252, 2021.
- [89] J. Mlejnek and J. Kubalik, “Evolutionary hyperheuristic for capacitated vehicle routing problem,” in *Proceedings of the 15th annual conference companion on Genetic and evolutionary computation*, 2013, pp. 219–220.
- [90] M. Okulewicz and J. Mańdziuk, “A particle swarm optimization hyper-heuristic for the dynamic vehicle routing problem,” *arXiv preprint arXiv:2006.08809*, 2020.
- [91] D. Meignan, A. Koukam, and J.-C. Créput, “Coalition-based metaheuristic: a self-adaptive metaheuristic using reinforcement learning and mimetism,” *Journal of Heuristics*, vol. 16, no. 6, pp. 859–879, 2010.
- [92] S. Asta and E. Özcan, “An apprenticeship learning hyper-heuristic for vehicle routing in hyflex,” in *2014 IEEE symposium on evolving and autonomous learning systems (EALS)*. IEEE, 2014, pp. 65–72.
- [93] R. Tyasnurita, E. Özcan, and R. John, “Learning heuristic selection using a time delay neural network for open vehicle routing,” in *2017 IEEE Congress on Evolutionary Computation (CEC)*. Ieee, 2017, pp. 1474–1481.
- [94] P. Kerschke, L. Kotthoff, J. Bossek, H. H. Hoos, and H. Trautmann, “Leveraging tsp solver complementarity through machine learning,” *Evolutionary computation*, vol. 26, no. 4, pp. 597–620, 2018.
- [95] K. Zhao, S. Liu, J. X. Yu, and Y. Rong, “Towards feature-free tsp solver selection: A deep learning approach,” in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.
- [96] A. E. Gutierrez-Rodríguez, S. E. Conant-Pablos, J. C. Ortiz-Bayliss, and H. Terashima-Marín, “Selecting meta-heuristics for solving vehicle routing problems with time windows via meta-learning,” *Expert Systems with Applications*, vol. 118, pp. 470–481, 2019.
- [97] S. Martin, D. Ouelhadj, P. Beullens, E. Ozcan, A. A. Juan, and E. K. Burke, “A multi-agent based cooperative approach to scheduling and routing,” *European Journal of Operational Research*, vol. 254, no. 1, pp. 169–178, 2016.
- [98] Z. Hlaing and M. A. Khine, “An ant colony optimization algorithm for solving traveling salesman problem,” in *International conference on information communication and management*, vol. 16, 2011, pp. 54–59.
- [99] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A critical

- review of recurrent neural networks for sequence learning,” *arXiv preprint arXiv:1506.00019*, 2015.
- [100] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities,” *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.
- [101] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [102] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.
- [103] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in neural information processing systems*, 2014, pp. 3104–3112.
- [104] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3128–3137.
- [105] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [106] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, “Neural combinatorial optimization with reinforcement learning,” *arXiv preprint arXiv:1611.09940*, 2016.
- [107] D. Levy and L. Wolf, “Learning to align the source code to the compiled object code,” in *International Conference on Machine Learning*. PMLR, 2017, pp. 2043–2051.
- [108] M. Deudon, P. Cournut, A. Lacoste, Y. Adulyasak, and L.-M. Rousseau, “Learning heuristics for the tsp by policy gradient,” in *International conference on the integration of constraint programming, artificial intelligence, and operations research*. Springer, 2018, pp. 170–181.
- [109] K. Li, T. Zhang, and R. Wang, “Deep reinforcement learning for multiobjective optimization,” *IEEE transactions on cybernetics*, vol. 51, no. 6, pp. 3103–3114, 2020.
- [110] Y. Kaempfer and L. Wolf, “Learning the multiple traveling salesmen problem with permutation invariant pooling networks,” *arXiv preprint arXiv:1803.09621*, 2018.
- [111] A. V. Le, P. Veerajagadheswar, P. Thiha Kyaw, M. R. Elara, and N. H. K. Nhan, “Coverage path planning using reinforcement learning-based tsp for hte-tran—a polyabolo-inspired self-reconfigurable tiling robot,” *Sensors*, vol. 21, no. 8, p. 2577, 2021.
- [112] Q. Ma, S. Ge, D. He, D. Thaker, and I. Drori, “Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning,” *arXiv preprint arXiv:1911.04936*, 2019.
- [113] H. Dai, E. B. Khalil, Y. Zhang, B. Dilkina, and L. Song, “Learning combinatorial optimization algorithms over graphs,” *arXiv preprint arXiv:1704.01665*, 2017.
- [114] H. Dai, B. Dai, and L. Song, “Discriminative embeddings of latent variable models for structured data,” in *International conference on machine learning*. PMLR, 2016, pp. 2702–2711.
- [115] A. L. Ottoni, E. G. Nepomuceno, and M. S. de Oliveira, “A response surface model approach to parameter estimation of reinforcement learning for the travelling salesman problem,” *Journal of Control, Automation and Electrical Systems*, vol. 29, no. 3, pp. 350–359, 2018.
- [116] Z.-H. Fu, K.-B. Qiu, and H. Zha, “Generalize a small pre-trained model to arbitrarily large tsp instances,” *arXiv preprint arXiv:2012.10658*, 2020.
- [117] R. Zhang, A. Prokhorchuk, and J. Dauwels, “Deep reinforcement learning for traveling salesman problem with time windows and rejections,” in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.
- [118] W. Kool, H. Van Hoof, and M. Welling, “Attention, learn to solve routing problems!” *arXiv preprint arXiv:1803.08475*, 2018.
- [119] Z. Xing and S. Tu, “A graph neural network assisted monte carlo tree search approach to traveling salesman problem,” *IEEE Access*, vol. 8, pp. 108 418–108 428, 2020.
- [120] E. Groshev, A. Tamar, M. Goldstein, S. Srivastava, and P. Abbeel, “Learning generalized reactive policies using deep neural networks,” in *2018 AAAI Spring Symposium Series*, 2018.
- [121] C. K. Joshi, T. Laurent, and X. Bresson, “An efficient graph convolutional network technique for the travelling salesman problem,” *arXiv preprint arXiv:1906.01227*, 2019.
- [122] M. Prates, P. H. Avelar, H. Lemos, L. C. Lamb, and M. Y. Vardi, “Learning to solve np-complete problems: A graph neural network for decision tsp,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 4731–4738.
- [123] N. Sultana, J. Chan, A. K. Qin, and T. Sarwar, “Learning to optimise general tsp instances,” *arXiv preprint arXiv:2010.12214*, 2020.
- [124] C. K. Joshi, T. Laurent, and X. Bresson, “On learning paradigms for the travelling salesman problem,” *arXiv preprint arXiv:1910.07210*, 2019.
- [125] M. Nazari, A. Oroojlooy, L. V. Snyder, and M. Takáč, “Reinforcement learning for solving the vehicle routing problem,” *arXiv preprint arXiv:1802.04240*, 2018.
- [126] R. A. A. A. Ibrahim, N. Lo and J. Ishaya, “Capacitated vehicle routing problem,” *International Journal of Research-Granthaalayah*, 2019.
- [127] B. Peng, J. Wang, and Z. Zhang, “A deep reinforcement learning algorithm using dynamic attention model for vehicle routing problems,” in *International Symposium on Intelligence Computation and Applications*. Springer, 2019, pp. 636–650.
- [128] L. Duan, Y. Zhan, H. Hu, Y. Gong, J. Wei, X. Zhang, and Y. Xu, “Efficiently solving the practical vehicle routing problem: A novel joint learning approach,” in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 3054–3063.

- [129] J. M. Vera and A. G. Abad, "Deep reinforcement learning for routing a heterogeneous fleet of vehicles," in *2019 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. IEEE, 2019, pp. 1–6.
- [130] J. K. Falkner and L. Schmidt-Thieme, "Learning to solve vehicle routing problems with time windows through joint attention," *arXiv preprint arXiv:2006.09100*, 2020.
- [131] L. Xin, W. Song, Z. Cao, and J. Zhang, "Step-wise deep learning models for solving routing problems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4861–4871, 2020.
- [132] —, "Multi-decoder attention model with embedding glimpse for solving vehicle routing problems," in *Proceedings of 35th AAAI Conference on Artificial Intelligence*, 2021.
- [133] K. Zhang, F. He, Z. Zhang, X. Lin, and M. Li, "Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 121, p. 102861, 2020.
- [134] J. Zhao, M. Mao, X. Zhao, and J. Zou, "A hybrid of deep reinforcement learning and local search for the vehicle routing problems," *IEEE Transactions on Intelligent Transportation Systems*, 2020.
- [135] N. Sultana, J. Chan, A. K. Qin, and T. Sarwar, "Learning vehicle routing problems using policy optimisation," *arXiv preprint arXiv:2012.13269*, 2020.
- [136] I. Drori, A. Kharkar, W. R. Sickinger, B. Kates, Q. Ma, S. Ge, E. Dolev, B. Dietrich, D. P. Williamson, and M. Udell, "Learning to solve combinatorial optimization problems on real-world graphs in linear time," in *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2020, pp. 19–24.
- [137] G. Bono, J. S. Dibangoye, O. Simonin, L. Matignon, and F. Pereyron, "Solving multi-agent routing problems using deep attention mechanisms," *IEEE Transactions on Intelligent Transportation Systems*, 2020.
- [138] B. Lin, B. Ghaddar, and J. Nathwani, "Deep reinforcement learning for the electric vehicle routing problem with time windows," *IEEE Transactions on Intelligent Transportation Systems*, 2021.
- [139] J. Li, L. Xin, Z. Cao, A. Lim, W. Song, and J. Zhang, "Heterogeneous attentions for solving pickup and delivery problem via deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, 2021.
- [140] J. Li, Y. Ma, R. Gao, Z. Cao, A. Lim, W. Song, and J. Zhang, "Deep reinforcement learning for solving the heterogeneous capacitated vehicle routing problem," *IEEE Transactions on Cybernetics*, 2021.
- [141] J. Li, D. Fu, Q. Yuan, H. Zhang, K. Chen, S. Yang, and F. Yang, "A traffic prediction enabled double rewarded value iteration network for route planning," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4170–4181, 2019.
- [142] J. James, W. Yu, and J. Gu, "Online vehicle routing with neural combinatorial optimization and deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3806–3817, 2019.
- [143] B. Balaji, J. Bell-Masterson, E. Bilgin, A. Damianou, P. M. Garcia, A. Jain, R. Luo, A. Maggari, B. Narayanaswamy, and C. Ye, "Orl: Reinforcement learning benchmarks for online stochastic optimization problems," *arXiv preprint arXiv:1911.10641*, 2019.
- [144] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.
- [145] D. Mukhutdinov, A. Filchenkov, A. Shalyto, and V. Vyatkin, "Multi-agent deep learning for simultaneous optimization for time and energy in distributed routing system," *Future Generation Computer Systems*, vol. 94, pp. 587–600, 2019.
- [146] R. Ramachandran Pillai and M. Arock, "An adaptive spiking neural p system for solving vehicle routing problems," *Arabian Journal for Science and Engineering*, vol. 45, no. 4, pp. 2513–2529, 2020.
- [147] Y. Sheng, H. Ma, and W. Xia, "A pointer neural network for the vehicle routing problem with task priority and limited resources," *Information Technology and Control*, vol. 49, no. 2, pp. 237–248, 2020.
- [148] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.
- [149] Z. Cao, H. Guo, W. Song, K. Gao, Z. Chen, L. Zhang, and X. Zhang, "Using reinforcement learning to minimize the probability of delay occurrence in transportation," *IEEE transactions on vehicular technology*, vol. 69, no. 3, pp. 2424–2436, 2020.
- [150] C. Chen, J. Jiang, N. Lv, and S. Li, "An intelligent path planning scheme of autonomous vehicles platoon using deep reinforcement learning on network edge," *IEEE Access*, vol. 8, pp. 99 059–99 069, 2020.
- [151] A. K. Kalakanti, S. Verma, T. Paul, and T. Yoshida, "RI solver pro: Reinforcement learning for solving vehicle routing problem," in *2019 1st International Conference on Artificial Intelligence and Data Sciences (AiDAS)*. IEEE, 2019, pp. 94–99.
- [152] J. Holler, R. Vuorio, Z. Qin, X. Tang, Y. Jiao, T. Jin, S. Singh, C. Wang, and J. Ye, "Deep reinforcement learning for multi-driver vehicle dispatching and repositioning problem," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 1090–1095.
- [153] N. Secomandi, "Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands," *Computers & Operations Research*, vol. 27, no. 11–12, pp. 1201–1225, 2000.
- [154] N. Vesselinova, R. Steinert, D. F. Perez-Ramirez, and M. Boman, "Learning combinatorial optimization on graphs: A survey with applications to networking," *IEEE Access*, vol. 8, pp. 120 388–120 416, 2020.

- [155] A. Subramanian, E. Uchoa, and L. S. Ochi, "A hybrid algorithm for a class of vehicle routing problems," *Computers & Operations Research*, vol. 40, no. 10, pp. 2519–2531, 2013.
- [156] X. Yao and Y. Liu, "Machine learning," in *Search Methodologies*. Springer, 2014, pp. 477–517.
- [157] N. Altman and M. Krzywinski, "The curse (s) of dimensionality," *Nat Methods*, vol. 15, no. 6, pp. 399–400, 2018.
- [158] M. Nickel, K. Murphy, V. Tresp, and E. Gaborilovich, "A review of relational machine learning for knowledge graphs," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 11–33, 2015.
- [159] K. T. Butler, D. W. Davies, H. Cartwright, O. Isayev, and A. Walsh, "Machine learning for molecular and materials science," *Nature*, vol. 559, no. 7715, pp. 547–555, 2018.
- [160] Q. Cappart, D. Chételat, E. Khalil, A. Lodi, C. Morris, and P. Veličković, "Combinatorial optimization and reasoning with graph neural networks," *arXiv preprint arXiv:2102.09544*, 2021.
- [161] K. Wagstaff, "Machine learning that matters," *arXiv preprint arXiv:1206.4656*, 2012.



Yongming He received the B.S. degree in logistics engineering from Chang'an University, Xi'an, China, in 2014.

He is currently pursuing the Ph.D. degree in management science and engineering with the College of Systems Engineering, National University of Defense Technology, Changsha, China. He was a visiting Ph.D. student with the University of Alberta, Edmonton, AB, Canada, from November 2018 to November 2019. His research interests include operations research, artificial intelligence, intelligent decision, task scheduling, and planning.



Mingfeng Fan received the B.S. degree in Transport Equipment and Control Engineering from Central South University, Changsha, China, in 2019. She is currently pursuing the Ph.D. degree in Traffic and Transportation Engineering with Central South University, Changsha, China. Her research interests include machine learning and UAV path planning.



Bingjie Li received her B.E. degree from Central South University, China, in 2019. Currently, she is working toward her M.E. degree at the School of Traffic & Transportation Engineering, Central South University. Her research interests include machine learning and path planning.



Guohua Wu received the B.S. degree in Information Systems and Ph.D degree in Operations Research from National University of Defense Technology, China, in 2008 and 2014, respectively. During 2012 and 2014, he was a visiting Ph.D student at University of Alberta, Edmonton, Canada. He is currently a Professor at the School of Traffic and Transportation Engineering, Central South University, Changsha, China. His current research interests include Planning and Scheduling, Computational Intelligence and Machine Learning. He has authored more than

80 referred papers including those published in IEEE TCYB, IEEE TSMCA and IEEE TITS. He serves as an Associate Editor of Swarm and Evolutionary Computation Journal, an editorial board member of International Journal of Bio-Inspired Computation, and a Guest Editor of Information Sciences and Memetic Computing. He is a regular reviewer of more than 20 journals including IEEE TEVC, IEEE TCYB, IEEE TSMCA and Information Sciences.



Witold Pedrycz is a Professor and the Canada Research Chair (CRC-Computational Intelligence) with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada, and also with the Department of Electrical and Computer Engineering, Faculty of Engineering, King Abdulaziz University, Jeddah, Saudi Arabia. He is also with the Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland. In 2012 he was elected a Fellow of the Royal Society of Canada.

TABLE VII
 REVIEWED PAPERS ABOUT LEARNING-BASED OPTIMIZATION ALGORITHMS.

year	Reference	Problem	Reference types		Approaches		Objectives		Benchmarks		Learning manner		Baselines			
			Conference	Journal	E2E	SbS	Length	Time	Cost	Simulation	Real data	SL	RL	Solver	Heuristics	LBO
	% of papers		32.90%	67.10%	65.70%	34.30%	75.70%	8.60%	21.40%	51.40%	60.00%	32.90%	70.00%	52.90%	75.70%	45.7%
2007	Lima <i>et al.</i> [61]	TSP	✓			✓	✓			✓	✓	✓	✓	✓	✓	
2009	Liu <i>et al.</i> [63]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
2010	Reinaldo <i>et al.</i> [76]	TSP	✓			✓	✓			✓	✓	✓	✓	✓	✓	✓
2014	Meignan <i>et al.</i> [91]	CVRPDVRP	✓	✓		✓	✓	✓		✓	✓	✓	✓	✓	✓	✓
2014	Asta and Özcan [92]	VRPTW				✓	✓			✓	✓	✓	✓	✓	✓	✓
2015	Vinyals <i>et al.</i> [85]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
2015	Bello <i>et al.</i> [106]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
2016	Martin <i>et al.</i> [97]	CVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Dai <i>et al.</i> [113]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
2017	Levy and Wolf [107]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Cooray and Rupasinghe [67]	EMVRP		✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Tyasanurita <i>et al.</i> [93]	OVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Deudon <i>et al.</i> [108]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Kaempfer and Wolf [110]	MTSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Groshev <i>et al.</i> [120]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Ottomi <i>et al.</i> [115]	TSP,ATSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
2018	Al-Duoli <i>et al.</i> [68]	CVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Alipour <i>et al.</i> [62]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Phiboonbanakit <i>et al.</i> [64]	CVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Kerschke <i>et al.</i> [94]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Nazari <i>et al.</i> [125]	CVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Kool <i>et al.</i> [118]	CVRP, TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Yao <i>et al.</i> [80]	Multi-objective route planning	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Yang <i>et al.</i> [77]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Chen and Tian [81]	CVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Joshi <i>et al.</i> [121]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Prates <i>et al.</i> [122]	TSP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Kalakanti <i>et al.</i> [151]	SDVRP, VRPTW	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Holler <i>et al.</i> [152]	MDVDRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Mukhtidinov <i>et al.</i> [145]	packet routing problem	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
2019	Ma <i>et al.</i> [112]	TSP, TSPTW	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Li <i>et al.</i> [141]	Online route planning	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Balaji <i>et al.</i> [143]	SVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Yu <i>et al.</i> [142]	Online route planning	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Vera and Abad [129]	CMVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Peng <i>et al.</i> [127]	CVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Hottung and Tierney [72]	CVRP, SDVRP	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓
	Rodriguez <i>et al.</i> [96]	VRPTW	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓

year	Reference	Problem	Reference types		Approaches		Objective		Benchmarks			Learning manner			Baselines		
			Conference	Journal	E2E	SbS	Length	Time	Cost	Simulation	Real data	SL	RL	Solver	Heuristics	LBO	
	% of papers		32.90	% 67.10%	65.70%	34.30%	75.70%	8.60%	21.40%	51.40%	60.00%	32.90%	70.00%	52.90%	75.70%	45.7%	
	Lu <i>et al.</i> [82]	CVRP	✓			✓	✓			✓		✓		✓		✓	
	Li <i>et al.</i> [109]	MOTSP		✓	✓		✓				✓	✓		✓		✓	
	Sultana <i>et al.</i> [123]	TSP		✓	✓		✓				✓	✓		✓		✓	
	Fu <i>et al.</i> [116]	TSP		✓	✓		✓				✓	✓		✓		✓	
	Zhanget <i>al.</i> [117]	TSPTWR		✓	✓		✓	✓			✓	✓		✓		✓	
	Xin <i>et al.</i> [131]	CVRP,TSP		✓	✓		✓				✓	✓		✓		✓	
	Xing <i>et al.</i> [119]	CVRP,TSP		✓	✓		✓				✓	✓		✓		✓	
	Zhao <i>et al.</i> [134]	CVRP, VRPTW		✓	✓		✓				✓	✓		✓		✓	
	Sultana <i>et al.</i> [135]	CVRP, TSP		✓	✓		✓				✓	✓		✓		✓	
	Drori <i>et al.</i> [136]	CVRP, TSP	✓		✓		✓				✓	✓		✓		✓	
	Bono <i>et al.</i> [137]	DS-CVRP, DS-CVRPTW		✓	✓		✓				✓	✓		✓		✓	
	Cao <i>et al.</i> [149]	SSP		✓	✓		✓	✓			✓	✓		✓		✓	
2020	Chen <i>et al.</i> [150]	CVRP		✓	✓		✓		✓		✓	✓		✓		✓	
	Ding <i>et al.</i> [65]	CVRP,TSP	✓		✓		✓		✓		✓	✓		✓		✓	
	Zhao <i>et al.</i> [95]	TSP		✓	✓		✓				✓	✓		✓		✓	
	Vlastelica <i>et al.</i> [86]	TSP	✓		✓		✓				✓	✓		✓		✓	
	Delarue <i>et al.</i> [79]	CVRP		✓	✓		✓				✓	✓		✓		✓	
	Sheng <i>et al.</i> [147]	VRPTRLR		✓	✓		✓		✓		✓	✓		✓		✓	
	Duan <i>et al.</i> [128]	CVRP	✓		✓		✓		✓		✓	✓		✓		✓	
	Zhang <i>et al.</i> [133]	MVRPSTW		✓	✓		✓		✓		✓	✓		✓		✓	
	Lin <i>et al.</i> [138]	EVPTW		✓	✓		✓		✓		✓	✓		✓		✓	
	Falkner and Schmidt-Thieme [130]	CVRPTW		✓	✓		✓		✓		✓	✓		✓		✓	
	Chen <i>et al.</i> [73]	CVRPTW		✓	✓		✓		✓		✓	✓		✓		✓	
	Joe and Lau [78]	DSVRP	✓		✓		✓		✓		✓	✓		✓		✓	
	Gao <i>et al.</i> [74]	CVRP, CVRPTW		✓	✓		✓		✓		✓	✓		✓		✓	
	Costa <i>et al.</i> [83]	TSP		✓	✓		✓		✓		✓	✓		✓		✓	
	Le <i>et al.</i> [111]	TSP		✓	✓		✓		✓		✓	✓		✓		✓	
	Xin <i>et al.</i> [132]	CVRP,TSP	✓		✓		✓		✓		✓	✓		✓		✓	
	Li <i>et al.</i> [139]	PDP		✓	✓		✓		✓		✓	✓		✓		✓	
2021	Lin <i>et al.</i> [138]	EVPTW		✓	✓		✓		✓		✓	✓		✓		✓	
	Sun <i>et al.</i> [66]	TSP		✓	✓		✓		✓		✓	✓		✓		✓	
	Wu <i>et al.</i> [84]	CVRP, TSP		✓	✓		✓		✓		✓	✓		✓		✓	