Revisiting Autoregressive GCNs for Vehicle Routing Problems

Zhipeng Zhong[1,*] **Junquan Huang**[1,2,*] **Yu Huang**[2] **Boyuan Zheng**[2]
Jungang Li[2] **Zihao Dongfang**[2] **Song Dai**[2] **Xuming Hu**[2] **Zong-Gan Chen**[1,†]
[1]School of Computer Science, South China Normal University
[2]AI Thrust, The Hong Kong University of Science and Technology (Guangzhou)

Abstract

Autoregressive (AR) Attention Models have become the dominant paradigm in neural solvers for vehicle routing problems (VRPs), while GCNs are almost exclusively paired with non-autoregressive (NAR) decoding and post-search algorithms. In this work, we revisit the role of decoding strategies and model properties in neural VRP solvers, and challenge the conventional pairing of NAR decoding with GCNs. We propose AR-GCN, an efficient and generalizable architecture for both Symmetric and Asymmetric VRPs (SVRPs and AVRPs). Extensive benchmarks across multiple tasks show that AR-GCN achieves advanced generalization performance, particularly on AVRPs (e.g., achieving an optimality gap of **1.64%** on ATSP-1K and **-5.29%** on ACVRP-1K). In particular, AR-GCN requires only **0.628M** learnable parameters and **3-epoch** training, providing new insights into the design of AR solvers for general VRPs. Our code and data are publicly available at https://github.com/HamJewel/AR-GCN-VRP.

1 Introduction

With the rapid development of Neural Combinatorial Optimization (NCO), various paradigms leveraging neural models have emerged to solve Vehicle Routing Problems (VRPs), such as the Traveling Salesman Problem (TSP) and the Capacitated Vehicle Routing Problem (CVRP). Heatmap-guided methods and constructive methods are two representative frameworks among the many solution paradigms. Specifically, heatmap-guided methods [1–3] generate edge-indicative heatmaps in a **non-autoregressive** (NAR) manner, and then leverage these heatmaps to guide post-search algorithms to obtain concrete route solutions. In contrast, constructive methods [4–6] adopt an **autoregressive** (AR) strategy to iteratively predict the next node, thereby progressively constructing a complete tour. Based on a synthesis of prior work, there exist conventional pairings between model architectures and decoding strategies: NAR decoding is commonly combined with Graph Convolutional Networks (GCNs), while AR decoding is typically paired with Attention Models. Adhering to these established configurations has left the performance potential of combining AR decoding with GCNs largely underexplored.

Revisiting Edge Awareness for Asymmetry Symmetric VRPs (SVRPs) in Euclidean space have been extensively studied over a long period of time. Recently, Asymmetric VRPs (AVRPs) have gained increasing interest in the field, as their formulations are closer to real-world scenarios and more challenging. SVRPs can be represented either by explicit node coordinates or by a distance matrix, whereas AVRPs can only be fully represented using a distance matrix. This also implies that node-aware models cannot solve AVRPs, whereas edge-aware models can handle both SVRPs

*Equal contribution: zhipeng@m.scnu.edu.cn, jewel.huang@connect.hkust-gz.edu.cn
†Corresponding author: charleszg@qq.com

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

and AVRPs. However, Vanilla Attention Models are node-aware in nature: they only encode node coordinates and therefore cannot handle AVRPs. Furthermore, Anisotropic GCNs with Gating Mechanism [7] are inherently edge-aware, yet they have not been well exploited on AVRPs. Pan et al. [8] propose an Asymmetric TSP solver using GCNs in heatmap-based (NAR) paradigms, but its performance is limited under NAR decoding. Overall, it is worthwhile to revisit and fully leverage the potential of Anisotropic GCNs for AVRPs.

Revisiting Dynamic Decision-Making Capability Our study reveals that different decoding strategies lead to variations in the dynamic decision-making capability of models. Specifically, when Greedy Search is used to obtain tour solutions, heatmap-guided (NAR) methods are more prone to getting stuck in local optima, whereas constructive (AR) methods exhibit better generalization performance. As shown in Figure 1, the cumulative optimality gap of AR-GCN and NAR-GCN on Symmetric TSPs evolves as the number of traversal steps increases. At each step, the gap is calculated based on the cumulative route length over all preceding traversal steps, reflecting the overall quality of the sequential decisions made up to that point. We observe that the gap of NAR-GCN gradually increases as the search progresses, with its growth rate becoming particularly pronounced in the later stages. In contrast, AR-GCN maintains a relatively stable optimality gap throughout the process. The reason is that NAR decoding generates a static heatmap in a single forward pass, lacking the ability to perform dynamic corrections during post-search. As a result, when combined with Greedy Search, errors accumulate progressively, thereby degrading solution quality. In contrast, AR-GCN continuously constructs, embeds, and encodes subgraphs during step-by-step prediction, adapting its predictions to the dynamically evolving routing state. Furthermore, NAR decoding heavily relies on post-search algorithms [9, 2, 10] to refine greedy solutions, which are often based on carefully hand-crafted designs, thereby limiting the flexibility of the solution process.

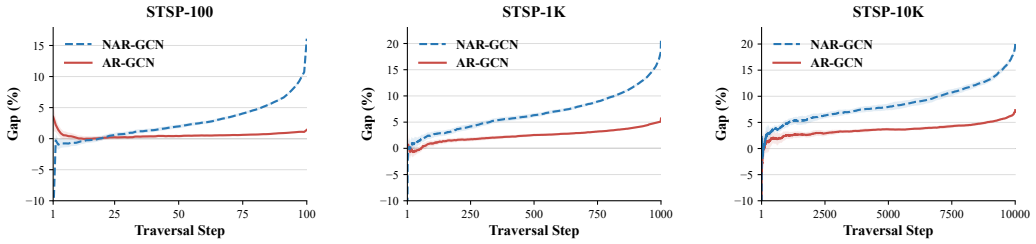


Figure 1: Cumulative optimality gap versus the number of traversal steps for AR-GCN and NAR-GCN.

Combining Anisotropic GCN with AR Decoding We revisit GCN-based routing solvers within an autoregressive framework. By synergizing the advantages of Anisotropic GCN and AR decoding, we propose **AutoRegressive GCN** (AR-GCN), an efficient and generalizable constructive method for solving SVRPs and AVRPs. In addition, we introduce **Min-Max Clamp** (MMC), a more universal edge normalization method that does not rely on explicit coordinates. MMC is designed to mitigate the influence of start-node outliers on normalization. Empirically, AR-GCN achieves lightweight and efficient networks while exhibiting superior cross-scale and cross-distribution generalization. In particular, our AR-GCN demonstrates remarkable training and parameter efficiency; it achieves superior performance on both SVRPs and AVRPs, with only 0.628M learnable parameters and 3-epoch training on 200K instances. We summarize our primary contributions as follows:

- We revisit the effects of AR and NAR decoding on GCNs for VRPs. We propose AR-GCN and demonstrate its superior dynamic decision-making capability and its intrinsic suitability to AVRPs, which are often overlooked by prior studies.
- Our AR-GCN exhibits high training efficiency, achieving rapid convergence within only 3-epoch training on 200K instances. Furthermore, AR-GCN contains only 0.628M learnable parameters, which is substantially fewer than those of competing baselines.
- We evaluate AR-GCN on both SVRPs and AVRPs, achieving advanced cross-scale and cross-distribution generalization. Our findings provide valuable insights into the design of edge-aware models for AVRPs.

2 Related Work

Solution Paradigms for VRPs The main solution paradigms of neural VRP solvers include constructive methods [4, 11], heatmap-guided methods [1, 2], divide-and-conquer methods [12, 13], and reconstruction-based methods [14, 15], among others. Among these solution paradigms, Attention Models [4] and Gated GCNs [1] constitute two classical and effective model backbones. Specifically, Attention Models integrate node features through Attention Mechanisms, whereas Gated GCNs leverage Gating Mechanism that projects edge features into gating scores, thereby modulating graph convolutions and integrating both node and edge features. As observed, NAR decoding is commonly adopted in heatmap-guided methods, whereas AR decoding is more widely used across different paradigms. Furthermore, Attention Models paired with AR decoding and Gated GCNs paired with NAR decoding represent conventional paradigms, whereas the potential of Gated GCNs paired with AR decoding remains underexplored.

Generalization in SVRPs Cross-scale and cross-distribution generalization are important objectives in related research, where input normalization plays a key role in achieving them. INViT [16] and RsGCN [10] propose the Invariant Nested View and Uniform Unit Square Projection, respectively. The former normalizes the coordinates of subgraph nodes, while the latter normalizes the lengths of subgraph edges. Both methods first construct subgraphs based on k-Nearest Neighbors (k-NN), then compute normalization coefficients using coordinate statistics, and finally normalize the subgraphs. They greatly facilitate cross-scale and cross-distribution generalization for SVRPs; however, due to their reliance on coordinate statistics, neither method is applicable to AVRPs.

Extension to AVRPs Vanilla Attention Models [6, 16] rely solely on node coordinates as input and focus on undirected-structure features between nodes, making them ineffective for AVRPs. To address this limitation, MatNet [5] and GOAL [17] introduce Mixed Attention Mechanisms to incorporate directed edge features. Specifically, MatNet conducts low-dimensional fusion by linearly combining edge lengths with attention scores, whereas GOAL performs high-dimensional fusion through additive integration of node and edge features. Moreover, GREAT [18] proposes a Graph Edge Attention Network for information propagation along edges. Notably, the edge-level Gating Mechanism in Gated GCNs possesses an anisotropic modeling property, making it inherently suitable for AVRPs. MatDIFFNet [8] further leverages this property for Asymmetric TSPs, which adopts a generative decoder with NAR decoding. However, existing GCN-based solvers for AVRPs exhibit limited performance, and the suitability of their anisotropic property for AVRPs has not been fully exploited. Furthermore, the weak dynamic decision-making capability caused by NAR decoding has become a limiting factor for the performance of Gated GCNs on AVRPs.

3 Preliminaries

TSP and CVRP in two-dimensional (2D) space are classic combinatorial optimization problems. In this paper, we aim to minimize the total length of Hamiltonian cycles that visit each customer node exactly once. Let STSP/ATSP-n denote Symmetric/Asymmetric TSPs with n customer nodes. Similarly, let SCVRP/ACVRP-n denote Symmetric/Asymmetric CVRPs consisting of n customers and **one depot**. We define the node set as $\mathcal{X}_n = \{\mathrm{x}_i\}_{i=1}^{n}$ for TSPs and $\mathcal{X}_n = \{\mathrm{x}_i\}_{i=0}^{n}$ for CVRPs, where an additional x_0 denotes the depot for CVRPs. Furthermore, CVRP incorporates vehicle capacity and customer demands, imposing a constraint that the remaining vehicle capacity must not be less than the demand of the customer to be served.

For SVRPs, each node can be represented by specific 2D coordinates; however, this does not apply to AVRPs with non-Euclidean metrics. To provide a unified representation for both SVRPs and AVRPs, we employ the distance matrix $\boldsymbol{D}$ to characterize pairwise distances, where $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ for TSPs and $\boldsymbol{D} \in \mathbb{R}^{(n+1)\times(n+1)}$ for CVRPs. Given a tour π, its total length L is defined as Equation (1). Note that unlike a TSP tour which forms a single Hamiltonian cycle, a complete CVRP tour is composed of several Hamiltonian sub-tours, each starting from and ending at the depot to satisfy vehicle capacity constraints.

$$L(\pi, \boldsymbol{D}) = \sum_{i=1}^{|\pi|-1} D\left(\mathrm{x}_{\pi_i}, \mathrm{x}_{\pi_{i+1}}\right). \tag{1}$$

4 Methodology

4.1 Subgraph Embedding

At each step of the autoregressive prediction [11, 16], we construct a subgraph $\mathcal{S}$ based on the current state, where $\mathcal{S}$ is a subset of the global node set $\mathcal{X}$. Specifically, $\mathcal{S}_1$ denotes the starting node, $\mathcal{S}_2$ denotes the current node, and $\mathcal{S}_{3\sim(k+2)}$ denote the candidate nodes. Moreover, k-NN sparsification can be optionally enabled. When enabled, $\mathcal{S}_{3\sim(k+2)}$ denotes the k nearest unvisited nodes; otherwise, it denotes all unvisited nodes. Subsequently, we index the global distance matrix $\boldsymbol{D}$ using $\mathcal{S}$ to obtain the subgraph distance matrix $\boldsymbol{E} \in \mathbb{R}^{(k+2)\times(k+2)}$, where $E_{i,j}$ denotes the distance of $s_i \to s_j$. Furthermore, three independent tensors $\mathbf{P}_{1\sim3} \in \mathbb{R}^h$ are randomly initialized to represent the starting node, the current node, and the candidate nodes, respectively. Finally, the subgraph is embedded into the h-dimensional space as follows:

$$e^0_{i,j} = \mathbf{W}_1 \tilde{E}_{i,j}, \;\; x^0_1 = \mathbf{P}_1, \;\; x^0_2 = \mathbf{P}_2, \;\; x^0_{3:} = \mathbf{P}_3, \tag{2}$$

$$x^0_i = x^0_i + \begin{cases} \mathbf{W}_2 \dfrac{r}{c}, & i = 2, \\ \mathbf{W}_2 \dfrac{q_i}{c}, & \text{otherwise}, \end{cases} \tag{3}$$

where $e^0_{i,j}$ and x^0_i denote the edge $s_i \to s_j$ and node s_i embeddings, respectively. $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{h\times1}$ are learnable parameter matrices, and : denotes the slicing operation. Specifically, for CVRP, x^0_i $(i \neq 2)$ and x^0_2 are augmented with the demand embedding and the remaining capacity embedding, respectively, where r denotes the remaining vehicle capacity, q_i denotes the demand of node s_i, and c denotes the maximum vehicle capacity. Moreover, $\tilde{\boldsymbol{E}}$ is obtained by normalizing $\boldsymbol{E}$ as described in Section 4.2 to ensure stable representations.

4.2 Min-Max Clamp

Input normalization [10, 16, 13] is crucial for improving generalization performance. Meanwhile, outliers can adversely affect the quality of normalization. Specifically, as a tour extends, the current node may become increasingly distant from the starting node, giving rise to pronounced distance outliers. To address outlier starting nodes, INViT [16] proposes clipping the coordinates of the starting node, which relies on the specific coordinate values. To eliminate the dependence on explicit coordinates and enable general applicability to distance-matrix inputs, we introduce Min-Max Clamp as defined in Equations (4) to (6):

$$\texttt{Clamp:}\; E_{i,1} = \min\left(\max\left(E_{i,2:}\right), E_{i,1}\right), \; i \neq 1, \tag{4}$$

$$\texttt{Reset:}\; E_{1,:} = \mathbf{0}, \tag{5}$$

$$\texttt{Scale:}\; E_{i,j} = \frac{E_{i,j} - \min\left(\boldsymbol{E}\right)}{\max\left(\boldsymbol{E}\right) - \min\left(\boldsymbol{E}\right)}, \tag{6}$$

where $E_{i,1}$ denotes the edge length from a non-starting node to the starting node; $E_{i,2:}$ denotes the edge lengths among non-starting nodes; $E_{1,:}$ denotes the edge lengths from the starting node to other nodes. First, we apply `Clamp` to the edges $s_i \to s_1$. Notably, `Clamp` is an asymmetric operation, meaning that $E_{i,1}$ and $E_{1,i}$ do not affect each other, which accounts for the input of AVRPs. Next, we `Reset` the edges $s_1 \to s_i$ to zero. This is motivated by the fact that, in VRPs, the model primarily focuses on 1) edges from the current node to candidate nodes and 2) edges from candidate nodes back to the starting node, whereas 3) edges from the starting node to candidate nodes are **not** traversed in practice and thus have limited relevance. `Reset` can remove the bias introduced by $s_1 \to s_i$. Finally, we apply Min-Max `Scale` to the entire edge matrix, where $\min(\boldsymbol{E}) \equiv 0$ in practice.

4.3 Autoregressive GCN (AR-GCN)

Encoding To better represent AVRPs, AR-GCN employs dual node representations [5, 8]. Figure 2 illustrates the architecture of a single encoder layer, which consists of a shared edge convolution and two node convolutions corresponding to outgoing and incoming nodes (out-node and in-node), respectively. In detail, the ℓ-th ($\ell \in \mathbb{Z}^+$) layer encoder is formally defined in Equations (7) to (13):

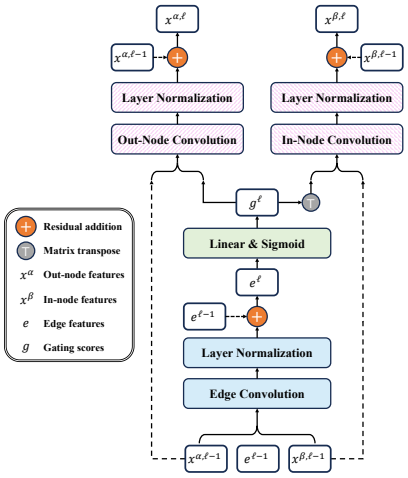


Figure 2: Overview of the encoder architecture in AR-GCN.

$$e_{i,j}^{\ell} = \mathbf{W}_3^{\ell} e_{i,j}^{\ell-1} \odot (\mathbf{W}_4^{\ell} x_i^{\alpha,\ell-1} + \mathbf{W}_5^{\ell} x_j^{\beta,\ell-1}), \tag{7}$$

$$e_{i,j}^{\ell} = e_{i,j}^{\ell-1} + \mathrm{LN}\left(e_{i,j}^{\ell}\right), \tag{8}$$

$$g_{i,j}^{\ell} = \sigma\left(\mathbf{W}_6^{\ell} e_{i,j}^{\ell}\right), \tag{9}$$

$$x_i^{\alpha,\ell} = \sum_{j=1}^{k+2} \left(\mathbf{W}_7^{\ell} x_j^{\alpha,\ell-1} \odot g_{i,j}^{\ell}\right) \oslash \left(\sum_{j=1}^{k+2} g_{i,j}^{\ell} + \epsilon\right), \tag{10}$$

$$x_i^{\beta,\ell} = \sum_{j=1}^{k+2} \left(\mathbf{W}_8^{\ell} x_j^{\beta,\ell-1} \odot g_{i,j}^{\ell,\top}\right) \oslash \left(\sum_{j=1}^{k+2} g_{i,j}^{\ell,\top} + \epsilon\right), \tag{11}$$

$$x_i^{\alpha,\ell} = x_i^{\alpha,\ell-1} + \mathrm{LN}\left(x_i^{\alpha,\ell}\right), \tag{12}$$

$$x_i^{\beta,\ell} = x_i^{\beta,\ell-1} + \mathrm{LN}\left(x_i^{\beta,\ell}\right), \tag{13}$$

where $e_{i,j}^{\ell}$ denotes the edge features; $g_{i,j}^{\ell}$ denotes the gating scores; $x_i^{\alpha,\ell}$ and $x_i^{\beta,\ell}$ denote the out-node and in-node features, respectively; $\mathbf{W}_{3\sim8}^{\ell} \in \mathbb{R}^{h\times h}$ are learnable parameter matrices; LN denotes Layer Normalization [19]; σ denotes the sigmoid function; $\top$ denotes the matrix transpose operation; $\odot$ and $\oslash$ denote element-wise multiplication and division, respectively; ϵ denotes a small constant added to prevent division by zero. Specifically, dual-node features are first integrated through edge convolution (Eq. 7). The resulting edge features are then projected into gating scores (Eq. 9), where $\mathbf{W}_6^{\ell}$ is used to prevent mean drift from causing gating scores to lose discriminative power. Finally, dual-node convolutions are performed independently with mean aggregation modulated by gating scores (Eqs. 10 and 11). Additionally, the dual-node features are initialized with the same embedding as $x_i^{\alpha,0} = x_i^{\beta,0} = x_i^0$.

Decoding The decoding logits are computed based on dual-node features, and the cross-entropy loss is subsequently applied to optimize the model, as formulated in Equations (14) and (15):

$$\hat{y}_i = \begin{cases} \mathrm{MLP}\left(x_i^{\alpha,l} \parallel x_i^{\beta,l}\right) : \mathbb{R}^{2h} \to \mathbb{R}^1, & i \notin \{1,2\} \\ -\infty, & \text{otherwise} \end{cases}, \tag{14}$$

$$\mathcal{L} = -\sum_{i=1}^{k+2} y_i \log \mathrm{softmax}\left(\hat{y}\right)_i, \tag{15}$$

where $\hat{y}_i$ denotes the logit of node s_i; $\parallel$ denotes the vector concatenation operation; MLP denotes a two-layer multilayer perceptron with GELU [20] as the activation function; y denotes a one-hot label vector with exactly one element equal to 1 and all others equal to 0. Further implementation details for CVRP are provided in Appendix A.

5 Experiment

5.1 Dataset

Synthetic Datasets We generate SVRPs under **uniform**, **clustered**, **explosion**, and **implosion** distributions following Fang et al. [16], and AVRPs under the **TMAT** distribution following Kwon et al. [5]. For SVRPs, the train set contains 200K uniform-distribution instances with n of **100**. For each distribution, the test set consists of 1K, 100, 20, and 10 instances with n of **100**, **1K**, **5K**, and **10K**, respectively. For TMAT-distributed AVRPs, the train set contains 200K instances with n of **100**, while the test set consists of 200, 100, 20, and 10 instances with n of **50**, **100**, **500**, and **1K**, respectively. For CVRPs, the maximum vehicle capacity c is set to **50**, and customer demands are randomly sampled from the integer range [**1**, **10**]. Additional details on data generation are presented in Appendix B.

Real-World Datasets We obtain STSP and ATSP test sets from TSPLIB [21], and SCVRP test sets from **Set X** [22] and **Set XL** [23] of CVRPLIB, while publicly available ACVRP test set is still lacking. For STSPs, the test set includes 77 instances with n ranging from 51 to 18,512. For ATSPs, the test set includes 27 instances with n ranging from 17 to 443. For SCVRPs, Set X contains 100 instances with 100–1,000 customers, and Set XL contains 100 instances with 1,047–10,000 customers. We directly use the best known solutions of these instances as the benchmark.

5.2 Training

Our hardware setup consists of an **NVIDIA RTX 4090 (24 GB)** GPU and an **Intel Xeon Platinum 8358P (2.60 GHz)** CPU. For each task setting, the model is trained for only **3** epochs using the Adam optimizer [24]. To stabilize training, we apply global gradient clipping with an L2-norm threshold of **1.0**. The batch size is set to **100**, and the learning rate is initialized at **5e-4** and decays to **0** following a cosine annealing schedule. For AR-GCN, the feature dimension h is set to **128**, and the number of encoder layers l is set to **6**. Moreover, each training epoch takes about 3 hours when k=50 and 5.5 hours otherwise. The training loss is visualized in Appendix C.

In particular, for SVRPs, we apply k-NN **sparsification** to improve generalization and efficiency. However, for TMAT-distributed AVRPs, we preserve a **full-graph** view, since directed edge lengths exhibit weak inter-edge correlations, resulting in a large discrepancy of patterns between local and global optima. Moreover, when applying MMC to AVRPs, we disable `Clamp` and `Reset` and retain only `Scale`, since the current version [5] of TMAT is derived from uniform distribution with additional constraints, containing no significant outliers.

5.3 Testing

Baselines We reproduce and compare against the following baselines: 1) **Classical Heuristic Solver**: LKH-3 [25], HGS [26]; 2) **Node-Aware Methods**: POMO [27], Omni-VRP [28], LEHD [6], ELG [29], INViT [16], SIGD [30]; and 3) **Edge-Aware Methods**: MatNet [5], ICAM [31], GREAT [18], MatPOENet [8], GOAL [17], RADAR [32], RRNCO [33]. All these neural methods are based on AR Attention Models. "**Node-Aware**" indicates that the method can only solve SVRPs, while "**Edge-Aware**" indicates that it can handle SVRPs and AVRPs. The number of learnable parameters of neural methods is shown in Figure 3. Note that MatPOENet (11.188M) is not included, as its substantially larger parameter count would compromise the clarity of the visualization. Clearly, AR-GCN is parameter-efficient, with a significantly smaller number of learnable parameters than other neural baselines.

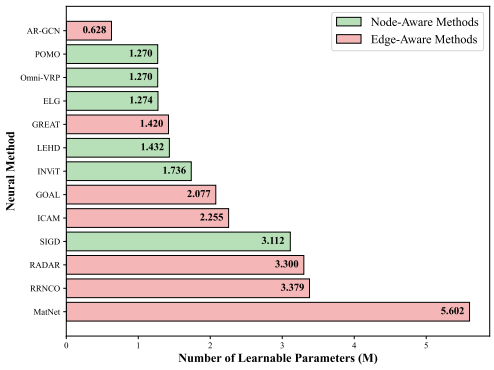


Figure 3: Learnable parameter comparison.

Inference For fairness, all neural baselines are tested using officially released checkpoints trained on instances with scales $\geq$ **100**. This setup ensures a clear evaluation of the generalization performance. For each instance, we generate **8** solutions for AR-GCN and all neural baselines (**6** for ATSP-1K and ACVRP-1K due to GPU memory limitations), except LEHD, for which we follow the original single-solution greedy setting. Subsequently, we select the best solution to compute metrics. For experimental alignment, we fix the inference batch size at **1**. Considering cost-effectiveness, the maximum runtime of LKH-3 and HGS is limited to n seconds. Moreover, the specific settings of LKH-3 and HGS are provided in Appendix B.

Metrics We adopt three widely used metrics: 1) **tour length**, 2) **optimality gap**, and 3) **solution time** per instance inference. The optimality gap is calculated as $(L/L^* - 1)$, where L^* denotes the benchmark tour length obtained by LKH-3 for TSPs and HGS for CVRPs. For TSPLIB and CVRPLIB, L^* is the best-known tour length. For each test set, we report the average of these metrics over all instances. Additional analysis of neural forward FLOPs is presented in Appendix E.

5.4 Main Results

For STSPs, as shown in Table 1, AR-GCN achieves the best overall performance, demonstrating strong cross-scale and cross-distribution generalization. For SCVRPs, as shown in Table 2, AR-GCN demonstrates robust competitiveness, particularly on the larger-scale instances of SCVRP-5K and SCVRP-10K. For ATSPs and ACVRPs, as shown in Table 3, AR-GCN achieves strong overall performance on ATSPs and ACVRPs, particularly on large-scale instances. Notably, even on ACVRP-500 and ACVRP-1K, it outperforms the state-of-the-art heuristic solver HGS. However, few existing neural baselines can effectively solve ACVRPs.

Table 1: Cross-scale and cross-distribution results on STSPs. (OOM indicates out-of-memory issues. For neural methods, the best and second-best optimality gaps are highlighted in **BOLD** and with UNDERLINE, respectively. BLUE , GREEN , and RED shading mark Classical Heuristic Solvers, Node-Aware Methods, and Edge-Aware Methods, respectively. "AR-GCN$_k$"/"AR-GCN" denote the variants trained and inferred with/without k-NN sparsification, respectively.)

Distribution	Uniform								Clustered							
Task	STSP-100		STSP-1K		STSP-5K		STSP-10K		STSP-100		STSP-1K		STSP-5K		STSP-10K	
Metrics	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time
LKH-3	0.00	2.54s	0.00	58.47s	0.00	9.81m	0.00	27.95m	0.00	5.65s	0.00	2.73m	0.00	28.62m	0.00	1.06h
POMO	0.35	0.05s	36.48	0.50s	OOM		OOM		4.15	0.05s	45.64	0.49s	OOM		OOM	
Omni-VRP	2.38	0.06s	21.39	0.55s	OOM		OOM		**1.81**	0.06s	20.21	0.55s	OOM		OOM	
LEHD	0.56	0.21s	3.15	2.41s	15.97	1.68m	26.07	11.35m	4.31	0.21s	13.84	2.42s	33.55	1.66m	50.84	11.25m
ELG	0.63	0.18s	14.49	1.79s	20.18	9.18s	OOM		3.89	0.18s	16.43	1.74s	24.26	9.01s	OOM	
INViT	2.78	0.51s	6.65	6.58s	7.95	49.36s	8.16	2.27m	5.15	0.51s	9.48	6.52s	9.80	49.00s	10.41	2.26m
SIGD	**0.04**	0.46s	**1.15**	7.94s	9.99	10.50m	OOM		2.52	0.44s	15.24	7.81s	106.09	10.52m	OOM	
ICAM	0.29	0.06s	2.67	0.49s	22.38	2.60s	26.00	7.82s	8.20	0.06s	18.31	0.50s	22.77	2.67s	23.20	7.86s
GREAT	3.41	0.11s	28.77	1.91s	OOM		OOM		13.99	0.10s	41.15	1.90s	OOM		OOM	
AR-GCN$_{50}$	0.14	0.31s	2.75	3.14s	5.71	14.45s	**6.12**	29.21s	3.04	0.31s	6.48	3.15s	7.13	14.48s	**7.14**	29.29s
AR-GCN$_{100}$	0.13	0.33s	2.12	5.99s	**5.54**	31.07s	6.28	1.04m	2.63	0.32s	**5.60**	5.99s	**6.71**	31.01s	7.24	1.04m

Distribution	Explosion								Implosion							
Task	STSP-100		STSP-1K		STSP-5K		STSP-10K		STSP-100		STSP-1K		STSP-5K		STSP-10K	
Metrics	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time
LKH-3	0.00	2.52s	0.00	1.13m	0.00	9.05m	0.00	29.52m	0.00	3.33s	0.00	2.78m	0.00	25.17m	0.00	1.10h
POMO	0.66	0.05s	42.95	0.48s	OOM		OOM		0.61	0.05s	36.77	0.48s	OOM		OOM	
Omni-VRP	2.42	0.06s	21.97	0.53s	OOM		OOM		2.38	0.06s	21.15	0.55s	OOM		OOM	
LEHD	0.70	0.21s	6.11	2.40s	21.51	1.66m	33.35	11.26m	1.21	0.21s	4.89	2.40s	18.25	1.67m	27.52	11.30m
ELG	0.91	0.18s	16.44	1.76s	23.72	9.09s	OOM		0.87	0.18s	14.26	1.75s	20.48	9.25s	OOM	
INViT	2.88	0.51s	9.56	6.53s	10.02	49.33s	10.72	2.27m	3.07	0.51s	7.39	6.55s	8.67	49.37s	8.66	2.27m
SIGD	**0.06**	0.41s	**2.82**	7.87s	45.06	10.51m	OOM		**0.28**	0.41s	3.37	7.86s	20.11	10.53m	OOM	
ICAM	1.53	0.06s	12.63	0.50s	20.29	2.61s	21.51	7.73s	0.69	0.06s	5.84	0.47s	22.97	2.71s	25.97	7.84s
GREAT	4.38	0.11s	30.33	1.90s	OOM		OOM		4.33	0.11s	28.95	1.87s	OOM		OOM	
AR-GCN$_{50}$	0.23	0.31s	4.14	3.24s	6.59	14.75s	7.88	29.73s	0.46	0.31s	3.80	3.15s	**5.96**	14.39s	**6.32**	29.13s
AR-GCN$_{100}$	0.13	0.33s	2.92	6.02s	**6.53**	31.24s	**7.64**	1.04m	0.34	0.33s	**2.88**	5.99s	**5.96**	31.07s	6.75	1.04m

Table 2: Cross-scale and cross-distribution results on SCVRPs.

Distribution	Uniform								Clustered							
Task	SCVRP-100		SCVRP-1K		SCVRP-5K		SCVRP-10K		SCVRP-100		SCVRP-1K		SCVRP-5K		SCVRP-10K	
Metrics	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time
HGS	0.00	1.67m	0.00	16.67m	0.00	1.39h	0.00	2.78h	0.00	1.67m	0.00	16.67m	0.00	1.39h	0.00	2.78h
LKH-3	0.53	45.29s	0.58	11.49m	6.82	1.40h	22.14	2.82h	0.22	50.25s	0.43	12.50m	5.89	1.40h	29.24	2.82h
POMO	2.61	0.09s	62.67	0.88s	OOM		OOM		3.30	0.09s	44.53	0.91s	OOM		OOM	
Omni-VRP	4.73	0.10s	8.54	0.87s	OOM		OOM		3.34	0.09s	4.90	0.92s	OOM		OOM	
LEHD	4.36	0.27s	2.98	2.94s	6.79	1.71m	13.30	11.37m	4.93	0.27s	10.10	2.95s	40.46	1.70m	34.68	11.36m
ELG	3.37	0.32s	6.43	3.59s	5.17	18.11s	OOM		4.56	0.36s	8.70	3.65s	13.11	19.94s	OOM	
INViT	6.84	1.08s	8.69	13.85s	5.39	1.66m	5.13	4.52m	6.94	1.11s	7.05	14.12s	5.69	1.68m	5.25	4.56m
SIGD	2.64	0.56s	5.06	14.49s	4.98	1.45m	6.70	2.88m	3.50	0.59s	9.08	14.63s	9.70	1.41m	9.08	2.98m
ICAM	3.39	0.10s	1.29	0.88s	3.68	4.80s	3.12	12.54s	5.01	0.10s	3.30	0.90s	3.71	4.73s	5.01	12.60s
GREAT	4.41	0.18s	10.67	3.41s	OOM		OOM		5.03	0.17s	9.48	3.53s	OOM		OOM	
GOAL	**1.54**	0.85s	**1.24**	11.06s	0.47	2.10m	OOM		**2.57**	0.85s	3.46	11.06s	1.48	2.10m	OOM	
AR-GCN$_{50}$	3.25	0.38s	2.27	3.68s	0.38	17.74s	0.45	37.94s	3.55	0.38s	2.10	3.86s	1.07	18.72s	0.84	39.43s
AR-GCN$_{100}$	2.53	0.40s	2.26	6.63s	**0.22**	34.02s	**0.00**	1.12m	3.43	0.40s	**2.01**	6.60s	**0.64**	33.62s	**0.53**	1.11m

Distribution	Explosion								Implosion							
Task	SCVRP-100		SCVRP-1K		SCVRP-5K		SCVRP-10K		SCVRP-100		SCVRP-1K		SCVRP-5K		SCVRP-10K	
Metrics	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time
HGS	0.00	1.67m	0.00	16.67m	0.00	1.39h	0.00	2.78h	0.00	1.67m	0.00	16.67m	0.00	1.39h	0.00	2.78h
LKH-3	0.53	45.47s	0.86	11.91m	8.58	1.40h	23.57	2.83h	0.53	45.64s	0.95	11.56m	7.71	1.40h	27.76	2.86h
POMO	2.81	0.09s	61.95	0.88s	OOM		OOM		2.80	0.09s	53.96	0.88s	OOM		OOM	
Omni-VRP	4.80	0.10s	9.05	0.91s	OOM		OOM		4.73	0.10s	8.27	0.92s	OOM		OOM	
LEHD	4.39	0.27s	5.26	2.97s	14.10	1.70m	22.26	11.35m	4.48	0.27s	8.34	2.98s	15.54	1.71m	20.21	11.37m
ELG	3.68	0.36s	7.18	3.55s	6.62	17.90s	OOM		3.53	0.36s	6.67	3.39s	6.15	16.54s	OOM	
INViT	7.44	1.12s	8.85	14.24s	5.69	1.68m	5.02	4.56m	7.07	1.12s	8.51	14.24s	5.55	1.68m	4.96	4.57m
SIGD	3.00	0.59s	6.17	13.92s	8.38	1.45m	11.53	3.01m	2.98	0.60s	6.12	14.67s	6.31	1.45m	6.91	2.97m
ICAM	3.88	0.10s	3.66	0.89s	4.56	4.91s	14.93	12.88s	3.69	0.10s	**2.04**	0.86s	2.47	4.60s	3.79	12.18s
GREAT	4.55	0.18s	14.38	3.51s	OOM		OOM		4.53	0.17s	11.16	3.41s	OOM		OOM	
GOAL	**1.66**	0.86s	**1.68**	11.12s	0.50	2.11m	OOM		**1.95**	0.85s	2.08	11.14s	0.87	2.10m	OOM	
AR-GCN$_{50}$	3.21	0.38s	2.44	3.70s	0.42	17.72s	0.50	37.82s	3.19	0.39s	2.58	3.86s	0.71	18.31s	0.49	39.55s
AR-GCN$_{100}$	2.71	0.41s	2.36	6.62s	**0.29**	33.95s	**-0.03**	1.12m	2.83	0.41s	2.57	6.61s	**0.49**	33.64s	**0.09**	1.12m

The results on real-world datasets for ATSPs, STSPs, and SCVRPs are presented in Tables 4 to 7. AR-GCN outperforms competing baselines overall on real-world ATSPs and STSPs, with particularly pronounced gains on ATSPs. Moreover, AR-GCN demonstrates competitive performance on real-world SCVRPs, although there is still room for further improvement. We conjecture that

this limitation may be related to experimental settings or the quality of the training data. Overall, the comprehensive performance of AR-GCN sufficiently demonstrates its architectural efficiency, particularly its suitability for AVRPs. The complete results are presented in Appendix F.

Table 3: Cross-scale results on TMAT-distributed ATSPs and ACVRPs. (— indicates issues of inapplicability or reproducibility.)

Distribution	TMAT															
Task	**ATSP-50**		**ATSP-100**		**ATSP-500**		**ATSP-1K**		**ACVRP-50**		**ACVRP-100**		**ACVRP-500**		**ACVRP-1K**	
Metrics	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time	Gap(%)	Time
HGS	—		—		—		—		0.00	50s	0.00	1.67m	0.00	8.33m	0.00	16.67m
LKH-3	0.00	1.59s	0.00	5.09s	0.00	34.70s	0.00	1.25m	0.65	49.01s	2.75	1.82m	10.10	11.23m	9.71	26.23m
MatNet	2.54	0.03s	5.06	0.05s	—		—		—		—		—		—	
ICAM	15.44	0.03s	7.51	0.06s	11.59	0.26s	20.68	0.50s	—		—		—		—	
GREAT	6.77	0.08s	10.39	0.11s	28.82	0.77s	63.25	2.49s	8.26	0.11s	9.90	0.15s	215.06	1.31s	269.52	3.58s
MatPOENet	12.45	0.05s	29.08	0.08s	173.24	0.61s	263.41	1.86s	—		—		—		—	
GOAL	5.52	0.35s	3.53	0.73s	1.93	36.67s	2.24	3.64m	—		—		—		—	
RADAR	2.84	0.04s	0.90	0.07s	2.82	0.28s	5.76	0.57s	5.78	0.07s	**2.71**	0.11s	12.66	0.52s	17.61	1.02s
RRNCO	25.88	0.08s	35.15	0.14s	65.33	0.67s	80.24	1.44s	22.75	0.10s	22.96	0.18s	47.10	0.87s	57.25	1.77s
AR-GCN	**0.40**	0.15s	**0.89**	0.33s	**1.31**	29.35s	**1.64**	2.89m	**1.76**	0.18s	2.75	0.40s	**-1.83**	29.63s	**-5.29**	2.90m

Table 4: Average gaps (%) across scale ranges on ATSPs from TSPLIB. († indicates that there exist unsolved instances in a scale range due to OOM or inapplicability.)

Scale Range #Instances	≤50 8	(50, 100] 7	(100, 200] 8	>200 4	All 27
ICAM	—	50.27	54.40	77.17	57.67†
GREAT	8.25	16.28	38.60	321.65	65.75
MatPOENet	29.57	50.86	76.10	80.41	56.41
GOAL	23.24	10.80	8.89	87.76	25.32
RADAR	19.70	25.63	50.36	72.92	38.20
RRNCO	13.20	24.46	53.76	36.65	31.61
AR-GCN	**2.05**	**4.13**	**7.25**	**28.99**	**8.12**

Table 5: Average gaps (%) across scale ranges on STSPs from TSPLIB.

Scale Range #Instances	≤100 12	(100, 1K] 36	(1K, 10K] 24	>10K 5	All 77
POMO	1.64	13.65	52.19†	OOM	22.43†
Omni-VRP	3.06	6.25	32.86†	OOM	14.07†
LEHD	0.61	3.41	12.46	48.74†	7.12†
ELG	1.48	6.21	15.26†	OOM	7.93†
INViT	2.53	5.75	8.69	8.87	6.37
SIGD	0.25	2.96	14.03	OOM	6.20†
ICAM	4.19	7.86	15.82	21.38†	10.06†
GREAT	6.78	16.17	36.28†	OOM	19.69†
AR-GCN$_{50}$	0.20	2.96	6.37	**6.82**	3.84
AR-GCN$_{100}$	**0.11**	**2.46**	**5.77**	7.94	**3.48**

Table 6: Average gaps (%) across scale ranges on Set X of CVRPLIB.

Scale Range #Instances	[100, 200] 22	(200, 500] 46	(500, 1K] 32	All 100
POMO	9.05	18.28	46.09	25.15
Omni-VRP	7.22	7.08	9.09	7.76
LEHD	11.75	9.45	17.59	12.56
ELG	7.06	7.33	9.55	7.98
INViT	8.24	11.67	11.41	10.83
SIGD	26.40	13.35	19.15	18.08
ICAM	6.08	5.08	5.63	5.47
GREAT	14.64	13.26	37.29	21.25
GOAL	6.04	**4.44**	**5.38**	**5.09**
AR-GCN$_{50}$	**5.73**	5.43	6.38	5.80
AR-GCN$_{100}$	5.78	6.41	7.13	6.50

Table 7: Average gaps (%) across scale ranges on Set XL of CVRPLIB.

Scale Range #Instances	(1K, 2.5K] 32	(2.5K, 5K] 36	(5K, 7.5K] 18	>7.5K 14	All 100
POMO	167.77	253.22	331.76†	OOM	221.14†
Omni-VRP	16.77	22.43	32.92†	OOM	20.50†
LEHD	24.11	29.72	34.95	33.28	29.37
ELG	14.09	14.33†	OOM	OOM	14.21†
INViT	12.89	13.76	13.77	11.63	13.19
SIGD	17.09	18.22	18.43	20.18	18.17
ICAM	7.83	8.64	9.03	12.02	8.93
GREAT	43.86†	OOM	OOM	OOM	43.86†
GOAL	**6.46**	**6.21**	**6.58**	OOM	**6.38**†
AR-GCN$_{50}$	7.48	7.08	6.74	6.61	7.08
AR-GCN$_{100}$	8.79	7.58	8.86	**6.49**	8.05

5.5 Ablation Study

AR-GCN vs. NAR GCNs To verify that AR-GCN exhibits stronger dynamic decision-making capability than NAR GCNs, we evaluate both methods without employing any hand-crafted post-search algorithms. We compare against two competitive NAR GCNs equipped with generative decoders, Fast T2T [34] and MatDIFFNet [8]. In addition, we allow Fast T2T to enable gradient search, while MatDIFFNet employs 2-Opt [35] to further improve greedy solutions. The ablation results are reported in Table 8. Evidently, AR-GCN achieves stronger generalization on both STSPs and ATSPs with substantially lower GPU memory consumption, with the advantage becoming more pronounced as the problem scale increases. However, even when Fast T2T is equipped with gradient search and further fine-tuned, its cross-distribution generalization remains poor. Moreover, gradient search significantly increases memory usage, and post-training fine-tuning is also computationally expensive for NAR GCNs. Meanwhile, MatDIFFNet exhibits poor cross-scale generalization on ATSPs, and even with 2-Opt enabled, the resulting improvement is limited. Furthermore, AR-GCN is highly parameter-efficient with only 0.628M learnable parameters, which is approximately one-tenth of Fast T2T (5.334M) and MatDIFFNet (6.986M).

Furthermore, to isolate the effect of different model backbones, we derive NAR-GCN from AR-GCN by modifying only its decoding and training paradigms, while keeping the model parameters and datasets identical. Both AR-GCN and NAR-GCN use a receptive field k of 50 and a solution count of 8. Since NAR-GCN relies on static heatmaps for search, it struggles to accommodate dynamic capacity constraints and depot-return decisions, making it unsuitable for CVRPs. As shown in Table 9, AR-GCN significantly outperforms NAR-GCN on STSPs across distributions and scales.

Single-Node vs. Dual-Node Representation Dual-node representation effectively captures asymmetric features and plays a crucial role in solving AVRPs. To validate its effectiveness, we evaluate AR-GCN with single-node and dual-node representations, while keeping all other settings consistent with the main experiments. As shown in Table 10, dual-node representation achieves lower optimality gaps than single-node representation on both ATSPs and ACVRPs across all scales.

Table 8: Ablation results of AR-GCN vs. NAR GCNs. ("Mem" denotes the maximum GPU memory usage; T_s denotes the number of inference steps; T_g denotes the number of gradient search steps; N_g denotes the number of greedy-search solutions; "+ 2-Opt" indicates that 2-Opt is enabled for post-optimization. Fast T2T adopts k-NN sparsification, with k set to 100 following the default setting. Notably, the plain "Fast T2T" denotes the weights trained only on STSP-100 with uniform distribution, whereas "Fast T2T*" denotes the weights fine-tuned on STSP-1K and STSP-10K. For MatDIFFNet, we use the weights trained on TMAT-distributed ATSP-100.)

Distribution	Uniform						Clustered					
Task	STSP-1K			STSP-10K			STSP-1K			STSP-10K		
Metrics	Gap(%)	Mem(GB)	Time	Gap(%)	Mem(GB)	Time	Gap(%)	Mem(GB)	Time	Gap(%)	Mem(GB)	Time
Fast T2T (T_s=5, T_g=0)	27.74	0.80	0.77s	101.75	7.81	62.48s	62.68	0.80	0.80s	123.19	7.81	61.52s
Fast T2T* (T_s=5, T_g=0)	6.41	0.80	0.73s	26.67	7.81	61.25s	21.20	0.80	0.78s	51.20	7.81	64.16s
AR-GCN$_{50}$ (N_g=1)	5.48	**0.02**	3.08s	**7.40**	**0.76**	30.46s	10.29	**0.02**	3.24s	**8.64**	**0.76**	32.96s
AR-GCN$_{100}$ (N_g=1)	**4.31**	0.05	3.08s	8.03	**0.76**	30.64s	**8.28**	0.05	3.11s	8.71	**0.76**	31.23s
Fast T2T (T_s=5, T_g=5)	22.66	11.12	5.17s		OOM		53.92	11.12	5.54s		OOM	
Fast T2T* (T_s=5, T_g=5)	3.62	11.12	4.75s		OOM		15.54	11.12	5.54s		OOM	
AR-GCN$_{50}$ (N_g=5)	3.17	**0.07**	3.24s	**6.30**	**2.25**	32.40s	7.00	**0.07**	3.25s	7.38	**2.25**	32.33s
AR-GCN$_{100}$ (N_g=5)	**2.41**	0.19	3.87s	6.85	**2.25**	39.52s	**5.99**	0.19	3.89s	**7.30**	**2.25**	39.51s

Distribution	TMAT											
Task	ATSP-50			ATSP-100			ATSP-500			ATSP-1K		
Metrics	Gap(%)	Mem(GB)	Time	Gap(%)	Mem(GB)	Time	Gap(%)	Mem(GB)	Time	Gap(%)	Mem(GB)	Time
MatDIFFNet	68.75	0.06	0.51s	20.04	0.13	0.47s	171.02	2.44	7.27s	177.54	9.63	29.92s
MatDIFFNet (+ 2-Opt)	14.43	0.06	0.51s	7.40	0.13	0.49s	44.68	2.44	9.05s	59.42	9.63	46.31s
AR-GCN (N_g=1)	**2.55**	**0.02**	0.15s	**3.26**	**0.04**	0.31s	**2.04**	**0.73**	4.43s	**2.20**	**2.89**	33.99s

Table 9: Ablation results (gap, %) of AR and NAR paradigms based on the same model backbone.

Distribution	Uniform				Clustered			
Task	STSP-100	STSP-1K	STSP-5K	STSP-10K	STSP-100	STSP-1K	STSP-5K	STSP-10K
NAR-GCN	9.85	18.18	19.01	19.46	13.78	16.33	18.31	18.84
AR-GCN	**0.14**	**2.75**	**5.71**	**6.12**	**3.04**	**6.48**	**7.13**	**7.14**
Distribution	Explosion				Implosion			
Task	STSP-100	STSP-1K	STSP-5K	STSP-10K	STSP-100	STSP-1K	STSP-5K	STSP-10K
NAR-GCN	11.32	19.06	19.74	19.33	11.31	18.40	19.27	19.42
AR-GCN	**0.23**	**4.14**	**6.59**	**7.88**	**0.46**	**3.80**	**5.96**	**6.32**

Table 10: Ablation results (gap, %) of node representations on ATSPs and ACVRPs.

TMAT	ATSP-50	ATSP-100	ATSP-500	ATSP-1K	ACVRP-50	ACVRP-100	ACVRP-500	ACVRP-1K
Single	1.05	1.56	3.31	6.75	4.00	5.59	1.84	-0.86
Dual	**0.40**	**0.89**	**1.31**	**1.64**	**1.76**	**2.75**	**-1.83**	**-5.29**

Effect of Model Backbones To investigate the impact of different model backbones within our framework, we replace the Gated GCN backbone of AR-GCN with the edge-aware Transformer backbones of MatNet [5] and ICAM [31], while keeping the AR pipeline, rollout strategy, MMC configuration, graph sparsification settings, datasets, and training procedure consistent. In detail, all models use a hidden dimension of 128, 6 encoder layers, and 3 training epochs. The Transformer backbones employ 8 attention heads and a feed-forward dimension of 512. As shown in Table 11, AR-GCN achieves the lowest optimality gaps across all evaluated scales on both TMAT-distributed ATSPs and ACVRPs, further demonstrating the effectiveness of the Gated GCN with dual-node representation in solving AVRPs.

Effect of k-NN on TMAT AVRPs As discussed in Section 5.2, under the TMAT distribution, directed edge lengths exhibit weak inter-edge correlations, which leads to a substantial discrepancy of patterns between local and global optima. Accordingly, we retain a full-graph view for TMAT AVRPs to achieve better performance. Here, we adopt k-NN sparsification to investigate the impact of incomplete graph visibility on solving TMAT AVRPs. Specifically, we set $k \in \{50, 100, 200\}$, and re-run inference on ATSPs and ACVRPs. As shown in Table 12, when the graph view does not cover the full graph, the optimality gap increases as the visibility is reduced. This demonstrates that TMAT AVRPs exhibit a substantial discrepancy between local and global optima, and that incomplete visibility degrades solution quality. However, full-graph inference inevitably incurs higher computational costs; therefore, developing low-loss graph sparsification for TMAT AVRPs is a promising direction for future work. In addition, the sensitivity to the training and inference choices of k-NN on SVRPs is reported in Appendix D.

Table 11: Ablation results (gap, %) of different model backbones based on our framework on ATSPs and ACVRPs.

TMAT	**ATSP-50**	**ATSP-100**	**ATSP-500**	**ATSP-1K**
MatNet	2.83	3.52	12.52	30.82
ICAM	1.32	1.98	12.72	50.28
AR-GCN	**0.40**	**0.89**	**1.31**	**1.64**
TMAT	**ACVRP-50**	**ACVRP-100**	**ACVRP-500**	**ACVRP-1K**
MatNet	7.40	8.49	6.02	6.50
ICAM	4.79	5.68	0.78	-2.63
AR-GCN	**1.76**	**2.75**	**-1.83**	**-5.29**

Table 12: Ablation results (gap, %) of k-NN sparsification on ATSPs and ACVRPs

TMAT	**ATSP-50**	**ATSP-100**	**ATSP-500**	**ATSP-1K**
AR-GCN	0.40	0.89	1.31	1.64
AR-GCN$_{200}$	0.40	0.89	9.12	16.68
AR-GCN$_{100}$	0.40	0.89	15.99	21.94
AR-GCN$_{50}$	0.40	5.43	21.67	26.93
TMAT	**ACVRP-50**	**ACVRP-100**	**ACVRP-500**	**ACVRP-1K**
AR-GCN	1.76	2.75	-1.83	-5.29
AR-GCN$_{200}$	1.76	2.75	0.89	-0.31
AR-GCN$_{100}$	1.76	2.75	5.25	4.33
AR-GCN$_{50}$	1.76	5.21	10.30	8.67

Effect of MMC on Generalization In this study, we investigate the impact of individual MMC operations on generalization. As presented in Table 13, for STSPs, `Clamp` and `Reset` both play an important role in cross-scale and cross-distribution generalization. In contrast, for ATSPs, as discussed in Section 5.2, `Clamp` and `Reset` have a limited impact on cross-scale generalization for TMAT-distributed AVRPs. In future work, it would be valuable to investigate additional distributions for AVRPs, particularly those in which 1) local graph patterns generalize well to the global structure, and 2) significant outliers are present.

Table 13: Ablation results (gap, %) of MMC. (C, R, and S denote `Clamp`, `Reset`, and `Scale`, respectively.)

Uniform	**STSP-100**	**STSP-1K**	**STSP-10K**
AR-GCN$_{50}$ w/ C-R-S	0.14	2.75	6.12
AR-GCN$_{50}$ w/ C-S	0.15	3.03	11.13
AR-GCN$_{50}$ w/ S	0.14	3.22	14.19
Clustered	**STSP-100**	**STSP-1K**	**STSP-10K**
AR-GCN$_{50}$ w/ C-R-S	3.04	6.48	7.14
AR-GCN$_{50}$ w/ C-S	2.73	9.69	70.24
AR-GCN$_{50}$ w/ S	2.91	8.48	15.57
TMAT	**ATSP-50**	**ATSP-100**	**ATSP-1K**
AR-GCN w/ C-R-S	0.48	0.82	1.70
AR-GCN w/ S	0.40	0.89	1.64

6 Conclusion

Anisotropic GCNs naturally possess edge awareness, making them particularly suitable for AVRPs. Compared to NAR decoding, AR decoding better enhances the dynamic decision-making capability of GCNs and mitigates suboptimal solutions induced by static heatmap-guided paradigms. Motivated by these findings, we propose AR-GCN by integrating Gated GCNs with AR decoding. Extensive evaluations demonstrate the efficiency and generalization capability of AR-GCN. In particular, AR-GCN achieves state-of-the-art overall performance, significantly outperforming existing neural baselines across various AVRPs. Overall, our work largely explores the capability of Gated GCNs for solving AVRPs, providing new insights into the design of edge-aware solvers for general VRPs.

Acknowledgments and Disclosure of Funding

Funding This work was supported in part by the National Natural Science Foundation of China under Grant 62206100, in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2024A1515011708, in part by the Guangzhou Basic and Applied Basic Research Foundation under Grant 2023A04J0319, and in part by the Youth S&T Talent Support Programme of Guangdong Provincial Association for Science and Technology under Grant SKXRC2026760.

Competing Interests The authors declare no competing interests.

References

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

[2] Zhanghua Fu, Kaibin Qiu, and Hongyuan Zha. Generalize a small pre-trained model to arbitrarily large TSP instances. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7474–7482, 2021.

[3] Zhiqing Sun and Yiming Yang. DIFUSCO: Graph-based diffusion solvers for combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, pages 3706–3731, 2023.

[4] Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=ByxBFsRqYm`.

[5] Yeong-Dae Kwon, Jinho Choo, Iljoo Yoon, Minah Park, Duwon Park, and Youngjune Gwon. Matrix encoding networks for neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 34, pages 5138–5149, 2021.

[6] Fu Luo, Xi Lin, Fei Liu, Qingfu Zhang, and Zhenkun Wang. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. In *Advances in Neural Information Processing Systems*, volume 36, pages 8845–8864, 2023.

[7] Xavier Bresson and Thomas Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017.

[8] Wenzheng Pan, Hao Xiong, Jiale Ma, Wentao Zhao, Yang Li, and Junchi Yan. UniCO: On unified combinatorial optimization via problem reduction to matrix-encoded general TSP. In *International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=yEwakMNIex`.

[9] Keld Helsgaun. An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, 2000.

[10] Junquan Huang, Zong-Gan Chen, Yuncheng Jiang, and Zhi-Hui Zhan. RsGCN: Subgraph-based rescaling enhances generalization of GCNs for solving traveling salesman problems. *arXiv preprint arXiv:2506.00533*, 2025.

[11] Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. BQ-NCO: Bisimulation quotienting for efficient neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, pages 77416–77429, 2023.

[12] Xuanhao Pan, Yan Jin, Yuandong Ding, Mingxiao Feng, Li Zhao, Lei Song, and Jiang Bian. H-TSP: Hierarchically solving the large-scale traveling salesman problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9345–9353, 2023.

[13] Haoran Ye, Jiarui Wang, Helan Liang, Zhiguang Cao, Yong Li, and Fanzhang Li. GLOP: Learning global partition and local construction for solving large-scale routing problems in real-time. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20284–20292, 2024.

[14] Fu Luo, Xi Lin, Yaoxin Wu, Zhenkun Wang, Xialiang Tong, Mingxuan Yuan, and Qingfu Zhang. Boosting neural combinatorial optimization for large-scale vehicle routing problems. In *International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=TbTJJNjumY`.

[15] Ke Li, Fei Liu, Zhengkun Wang, and Qingfu Zhang. Destroy and repair using hyper graphs for routing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18341–18349, 2025.

[16] Han Fang, Zhihao Song, Paul Weng, and Yutong Ban. INViT: A generalizable routing problem solver with invariant nested view transformer. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pages 12973–12992, 2024.

[17] Darko Drakulic, Sofia Michel, and Jean-Marc Andreoli. GOAL: A generalist combinatorial optimization agent learner. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=z2z9suDRjw.

[18] Attila Lischka, Filip Rydin, Jiaming Wu, Morteza Haghir Chehreghani, and Balázs Kulcsár. A GREAT architecture for edge-based graph problems like TSP. *arXiv preprint arXiv:2408.16717*, 2024.

[19] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[20] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

[21] Gerhard Reinelt. TSPLIB—A traveling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384, 1991.

[22] Eduardo Uchoa, Diego Pecin, Artur Pessoa, Marcus Poggi, Thibaut Vidal, and Anand Subramanian. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858, 2017.

[23] Eduardo Queiroga, Rafael Martinelli, Anand Subramanian, Eduardo Uchoa, and Thibaut Vidal. The XL instances for the capacitated vehicle routing problem. *arXiv preprint arXiv:2601.11467*, 2026.

[24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[25] Keld Helsgaun. An extension of the Lin-Kernighan-Helsgaun TSP solver for constrained traveling salesman and vehicle routing problems. Technical report, Roskilde University, 2017.

[26] Thibaut Vidal. Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. *Computers & Operations Research*, 140:105643, 2022.

[27] Yeong-Dae Kwon, Jinho Choo, Byoungjip Kim, Iljoo Yoon, Youngjune Gwon, and Seungjai Min. POMO: Policy optimization with multiple optima for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21188–21198, 2020.

[28] Jianan Zhou, Yaoxin Wu, Wen Song, Zhiguang Cao, and Jie Zhang. Towards omni-generalizable neural methods for vehicle routing problems. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 42769–42789, 2023.

[29] Chengrui Gao, Haopu Shang, Ke Xue, Dong Li, and Chao Qian. Towards generalizable neural solvers for vehicle routing problems via ensemble with transferrable local policy. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, pages 6914–6922, 2024.

[30] Jonathan Pirnay and Dominik G. Grimm. Self-improvement for neural combinatorial optimization: Sample without replacement, but improvement. In *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=agT8ojoH0X.

[31] Changliang Zhou, Xi Lin, Zhenkun Wang, Xialiang Tong, Mingxuan Yuan, and Qingfu Zhang. Instance-conditioned adaptation for large-scale generalization of neural routing solver. *arXiv preprint arXiv:2405.01906*, 2024.

[32] Hang Yi, Ziwei Huang, Yining Ma, and Zhiguang Cao. RADAR: Learning to route with asymmetry-aware distance representations. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=lWdxX5s9T1.

[33] Jiwoo Son, Zhikai Zhao, Federico Berto, Chuanbo Hua, Zhiguang Cao, Changhyun Kwon, and Jinkyoo Park. RRNCO: Towards real-world routing with neural combinatorial optimization. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=sKvo9ZZfpe.

[34] Yang Li, Jinpei Guo, Runzhong Wang, Hongyuan Zha, and Junchi Yan. Fast T2T: Optimization consistency speeds up diffusion-based training-to-testing solving for combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 37, pages 30179–30206, 2024.

[35] G. A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6: 791–812, 1958.

[36] David Applegate, Robert Bixby, Vasek Chvatal, and William Cook. Concorde TSP solver. `https://math.uwaterloo.ca/tsp/concorde`, 2006.

[37] Roy Jonker and Ton Volgenant. Transforming asymmetric into symmetric traveling salesman problems. *Operations Research Letters*, 2(4):161–163, 1983.

[38] Yang Wang, Ya-Hui Jia, Wei-Neng Chen, and Yi Mei. Distance-aware attention reshaping for enhancing generalization of neural solvers. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2025.

[39] Attila Lischka, Jiaming Wu, Rafael Basso, Morteza Haghir Chehreghani, and Balázs Kulcsár. Less is more - on the importance of sparsification for transformers and graph neural networks for TSP. *arXiv preprint arXiv:2403.17159*, 2024.

[40] Michael Li, Eric Bae, Christopher Haberland, and Natasha Jaques. Generative modeling for robust deep reinforcement learning on the traveling salesman problem. *arXiv preprint arXiv:2508.08718*, 2025.

[41] Junrui Wen, Yifei Li, Bart Selman, and Kun He. LocalEscaper: A weakly-supervised framework with regional reconstruction for scalable neural TSP solvers. *arXiv preprint arXiv:2502.12484*, 2025.

[42] Minsu Kim, Sanghyeok Choi, Jiwoo Son, Hyeonah Kim, Jinkyoo Park, and Yoshua Bengio. Ant colony sampling with GFlowNets for combinatorial optimization. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258, pages 469–477, 2025.

[43] Lu Duan, Yang Zhan, Haoyuan Hu, Yu Gong, Jiangwen Wei, Xiaodong Zhang, and Yinghui 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*, pages 3054–3063, 2020.

[44] Haoran Ye, Jiarui Wang, Zhiguang Cao, Helan Liang, and Yong Li. DeepACO: Neural-enhanced ant systems for combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, pages 43706–43728, 2023.

[45] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=rJXMpikCZ`.

[46] Jakob Bossek, Pascal Kerschke, Aneta Neumann, Markus Wagner, Frank Neumann, and Heike Trautmann. Evolving diverse TSP instances by means of novel and creative mutation operators. In *Proceedings of the 15th ACM/SIGEVO Conference on Foundations of Genetic Algorithms*, pages 58–71, 2019.

[47] Minsu Kim, Junyoung Park, and Jinkyoo Park. Sym-NCO: Leveraging symmetricity for neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 35, pages 1936–1949, 2022.

[48] Jill Cirasella, David S. Johnson, Lyle A. McGeoch, and Weixiong Zhang. The asymmetric traveling salesman problem: Algorithms, instance generators, and tests. In *Algorithm Engineering and Experimentation*, volume 2153, pages 32–59, 2001.

[49] Yimeng Min, Yiwei Bai, and Carla P Gomes. Unsupervised learning for solving the travelling salesman problem. In *Advances in Neural Information Processing Systems*, volume 36, pages 47264–47278, 2023.

[50] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[51] Jun-Quan Huang, Zhen-An Liang, Zong-Gan Chen, Jian-Yu Li, Hong Zhao, Yuncheng Jiang, Zhi-Hui Zhan, and Jun Zhang. Crowdsourcing logistics scheduling based on an ant colony system approach. In *2024 11th International Conference on Machine Intelligence Theory and Applications (MiTA)*, pages 1–8, 2024.

[52] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations*, 2014. URL https://openreview.net/forum?id=DQNsQf-UsoDBa.

[53] Yang Li, Jiale Ma, Wenzheng Pan, Runzhong Wang, Haoyu Geng, Nianzu Yang, and Junchi Yan. Unify ML4TSP: Drawing methodological principles for TSP and beyond from streamlined design space of learning and search. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=grU1VKEOLi.

[54] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

[55] André Hottung, Yeong-Dae Kwon, and Kevin Tierney. Efficient active search for combinatorial optimization problems. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nO5caZwFwYu.

[56] Chaitanya K. Joshi, Quentin Cappart, Louis-Martin Rousseau, and Thomas Laurent. Learning TSP requires rethinking generalization. In *International Conference on Principles and Practice of Constraint Programming*, volume 210, pages 33:1–33:21, 2021.

[57] Zeyang Zhang, Ziwei Zhang, Xin Wang, and Wenwu Zhu. Learning to solve travelling salesman problem with hardness-adaptive curriculum. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9136–9144, 2022.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[59] Ruizhong Qiu, Zhiqing Sun, and Yiming Yang. DIMES: A differentiable meta solver for combinatorial optimization problems. In *Advances in Neural Information Processing Systems*, volume 35, pages 25531–25546, 2022.

[60] Benjamin Hudson, Qingbiao Li, Matthew Malencia, and Amanda Prorok. Graph neural network guided local search for the traveling salesperson problem. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=ar92oEosBIg.

[61] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

[62] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. DropEdge: Towards deep graph convolutional networks on node classification. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Hkx1qkrKPr.

[63] Yang Li, Jinpei Guo, Runzhong Wang, and Junchi Yan. T2T: From distribution learning in training to gradient search in testing for combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, pages 50020–50040, 2023.

[64] Liang Xin, Wen Song, Zhiguang Cao, and Jie Zhang. NeuroLKH: Combining deep learning model with Lin-Kernighan-Helsgaun heuristic for solving the traveling salesman problem. In *Advances in Neural Information Processing Systems*, volume 34, pages 7472–7483, 2021.

[65] Shipei Zhou, Yuandong Ding, Chi Zhang, Zhiguang Cao, and Yan Jin. DualOpt: A dual divide-and-optimize algorithm for the large-scale traveling salesman problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 27178–27186, 2025.

[66] Yifan Xia, Xianliang Yang, Zichuan Liu, Zhihao Liu, Lei Song, and Jiang Bian. Position: Rethinking post-hoc search-based neural approaches for solving large-scale traveling salesman problems. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pages 54178–54190, 2024.

[67] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=Skq89Scxx`.

[68] Mingzhao Wang, You Zhou, Zhiguang Cao, Yubin Xiao, Xuan Wu, Wei Pang, Yuan Jiang, Hui Yang, Peng Zhao, and Yuanshu Li. An efficient diffusion-based non-autoregressive solver for traveling salesman problem. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025.

[69] Yan Jin, Yuandong Ding, Xuanhao Pan, Kun He, Li Zhao, Tao Qin, Lei Song, and Jiang Bian. Pointerformer: Deep reinforced multi-pointer transformer for the traveling salesman problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8132–8140, 2023.

[70] Yubin Xiao, Di Wang, Boyang Li, Mingzhao Wang, Xuan Wu, Changliang Zhou, and You Zhou. Distilling autoregressive models to obtain high-performance non-autoregressive solvers for vehicle routing problems with faster inference speed. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20274–20283, 2024.

[71] Yong Liang Goh, Zhiguang Cao, Yining Ma, Jianan Zhou, Mohammed Haroon Dupty, and Wee Sun Lee. SHIELD: Multi-task multi-distribution vehicle routing solver with sparsity and hierarchy. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pages 19679–19711, 2025.

A Implementation Details for CVRP

Following the implementation of Luo et al. [6], we represent a tour of CVRP-n as a 2D matrix $\mathcal{T} \in \mathbb{R}^{(n+1)\times 2}$. $\mathcal{T}_{i,0}$ denotes the index of the node to be visited at step i, while $\mathcal{T}_{i,1}$ is a binary indicator specifying whether the vehicle returns to the depot after visiting $\mathcal{T}_{i,0}$. Specifically, $\mathcal{T}_{i,1} = 1$ indicates that the vehicle immediately returns to the depot (indexed by 0) after visiting $\mathcal{T}_{i,0}$, whereas $\mathcal{T}_{i,1} = 0$ means that no return to the depot is required. An example of a 2D tour and its flattened representation for CVRP-5 is shown in Equation (16).

$$\begin{bmatrix} 0 & 2 & 1 & 3 & 5 & 4 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \xrightarrow{\text{flatten}} [\,0\ 2\ 0\ 1\ 3\ 5\ 0\ 4\ 0\,]. \tag{16}$$

This design facilitates structure alignment and enables efficient batch operations during both training and inference.

B Details of Data and Solution Generation

B.1 Data Generation

The generation of SVRPs with diverse distributions (uniform, clustered, explosion, and implosion) involves multiple parameter settings. We adopt the code released by Fang et al. [16] and follow their configurations. Detailed descriptions can be found in their publication and are omitted here for brevity.

B.2 Solution Generation

Our training sets are solved using LKH-3. For consistency, we set the `RUNS` parameter of LKH-3 to 1 and `MAX_TRIALS` to 10K. Specifically, `MAX_CANDIDATES` is set to 5 for SVRPs and 6 for AVRPs, respectively. In addition, following the official examples, we enable the `SPECIAL` option for CVRPs. To avoid excessive time consumption, we set `TIME_LIMIT` to n seconds, where n denotes the problem scale. Despite allowing several hours of solution time, infeasible solutions (i.e., customer demands exceeding the remaining vehicle capacity) may still occur for SCVRP-5K and SCVRP-10K. In such cases, we apply an additional correction by inserting a return-to-depot action before any delivery that would exceed the remaining vehicle capacity. For HGS, we adopt its Python wrapper, PyHygese, to generate benchmark results for the SCVRP and ACVRP test sets. All hyperparameters are kept at their default settings, with only the maximum runtime constrained to n seconds. In addition, HGS does not fully support asymmetric distance matrices. The generated tour lengths may be computed based on symmetric distances, and therefore require manual recalculation for correction. Specifically, the average tour lengths before and after correction, as well as the average correction ratios, are reported in Table 14.

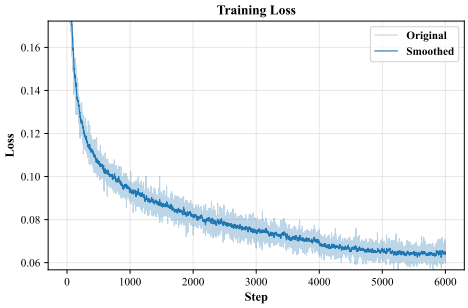
In AR-GCN, we adopt different rollout strategies for TSPs and CVRPs when generating multiple solutions. Specifically, when rolling out 8 solutions for TSPs, we randomly select 8 distinct nodes as starting nodes and perform greedy search from each of them. In contrast, for CVRPs, the starting node is fixed to the depot. Therefore, we initiate a beam search with width 8 from the depot, where the 8 beam nodes are selected from the candidate nodes with the highest probabilities. Subsequently, greedy search is performed independently from each of these beam nodes.

Table 14: Correction details for the tours generated by HGS and LKH-3. The result format is "pre-correction length → post-correction length (correction ratio)", where the correction ratio is computed as $(L_{\text{post}}/L_{\text{pre}} - 1)$.

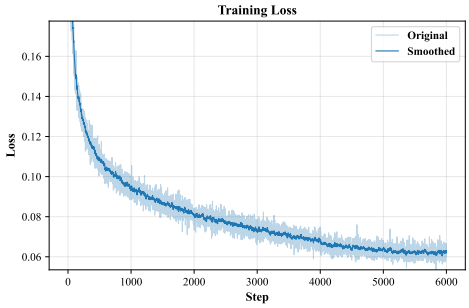
Task	**ACVRP-50**	**ACVRP-100**	**ACVRP-500**	**ACVRP-1K**
HGS (TMAT)	2.0218 → 2.0247 (0.14%)	2.1517 → 2.1575 (0.27%)	2.6416 → 2.6723 (1.16%)	2.9446 → 3.0084 (2.17%)
Task	**SCVRP-100**	**SCVRP-1K**	**SCVRP-5K**	**SCVRP-10K**
LKH-3 (Uniform)	16.77 → 16.77 (0.00%)	130.62 → 130.62 (0.00%)	685.44 → 685.44 (0.00%)	1383.11 → 1407.55 (1.77%)
LKH-3 (Clustered)	14.71 → 14.71 (0.00%)	115.42 → 115.42 (0.00%)	518.47 → 519.23 (0.15%)	1215.12 → 1267.29 (4.29%)
LKH-3 (Explosion)	14.25 → 14.25 (0.00%)	89.37 → 89.37 (0.00%)	396.34 → 396.68 (0.09%)	796.90 → 806.86 (1.25%)
LKH-3 (Implosion)	16.04 → 16.04 (0.00%)	122.07 → 122.07 (0.00%)	649.79 → 649.79 (0.00%)	1452.57 → 1492.34 (2.74%)

C Visualization of Training Loss

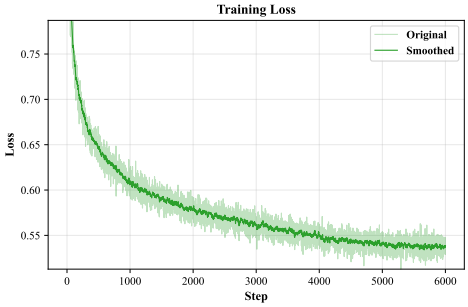
The training loss of AR-GCN on different tasks is shown in Figure 4. It can be observed that the training loss on AVRPs is higher than that on SVRPs. This can be attributed to the increased complexity of asymmetric features, which makes learning more challenging for the model.



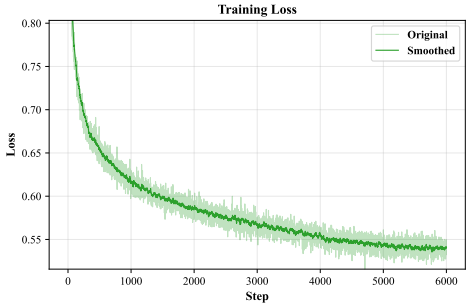
(a) AR-GCN$_{50}$ on STSP-100.



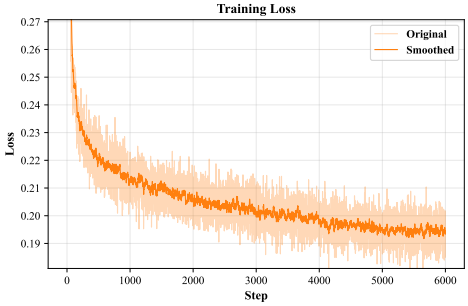
(b) AR-GCN$_{100}$ on STSP-100.



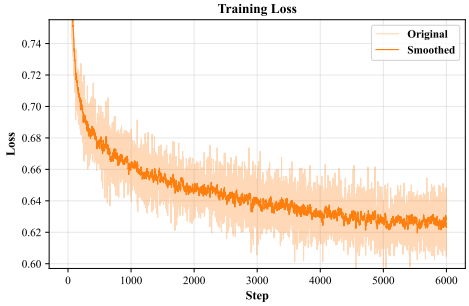
(c) AR-GCN$_{50}$ on SCVRP-100.



(d) AR-GCN$_{100}$ on SCVRP-100.



(e) AR-GCN on ATSP-100.



(f) AR-GCN on ACVRP-100.

Figure 4: Training losses of AR-GCN under different settings.

D Sensitivity to Receptive Field Size

Training time We train separate models with $k \in \{10, 25, 30, 50, 75, 100\}$ on uniform-distribution STSPs, and evaluate each model on STSPLIB using the same k as during training. As shown in Table 15, larger receptive fields generally lead to more robust performance on STSPs. However, on very large-scale instances (>10K), excessively large receptive fields do not necessarily provide further benefits and may even degrade performance.

Inference time We train the model with k=50 on uniformly distributed SCVRP instances, and evaluate it on both uniform- and clustered-distribution SCVRPs using $k \in \{30, 40, 50, 60, 70\}$. As shown in Table 16, when the inference-time k differs from the training setting, larger receptive fields

yield lower relative gaps on smaller-scale SCVRPs, whereas smaller receptive fields tend to perform better as the problem scale increases. On clustered distributions, which present more challenging spatial structures, smaller receptive fields can leverage spatial proximity to filter out low-quality candidate nodes, thereby reducing suboptimal decisions.

Table 15: Optimality gaps (%) on STSPLIB with matched training and inference k values.

k	≤100	(100, 1K]	(1K, 10K]	>10K	All
10	5.34	6.58	11.26	12.33	8.22
25	0.67	3.11	7.44	7.68	4.38
30	0.55	**2.40**	7.00	7.77	3.89
50	0.20	2.96	6.37	**6.82**	3.84
75	**0.08**	**2.40**	6.04	7.83	3.53
100	0.11	2.46	**5.77**	7.94	**3.48**

Table 16: Optimality gaps (%) on SCVRPs with varying inference k values (k=50 as the reference).

k	**Uniform**			**Clustered**		
	100	1K	5K	100	1K	5K
30	0.29	0.28	**-0.21**	0.23	-0.01	**-0.39**
40	0.02	0.02	-0.20	0.03	**-0.11**	-0.20
50	0.00	**0.00**	0.00	0.00	0.00	0.00
60	-0.11	0.14	0.23	-0.10	0.12	0.24
70	**-0.17**	0.18	0.47	**-0.10**	0.24	0.56

E Inference Computation

The main results show that AR-GCN achieves lower optimality gaps than several faster baselines. Node-aware models typically take $O(n)$ node features as input, whereas AR-GCN explicitly encodes pairwise relationships from an $O(n^2)$ distance matrix. Explicit edge encoding introduces additional neural computation, while solution time also reflects memory access and neighborhood selection. We therefore analyze FLOPs to quantify the computation involved in feature processing.

Following SHIELD [71], we report neural forward GFLOPs per instance in Table 17, complementing parameter counts and solution time. All methods use one rollout per instance. We count all neural forward operations in FP32, with one multiply–accumulate counted as two FLOPs. Input preparation and search bookkeeping are excluded.

For uniform STSP-10K, AR-GCN$_{50}$ requires 11.6 K GFLOPs per solution, compared with 502.0 for INViT. However, as shown in Table 1, AR-GCN$_{50}$ reduces solution time from 2.27m to 29.21s (about $4.7\times$ faster), while achieving a 25.0% lower optimality gap (6.12% vs. 8.16%). INViT's low neural FLOPs reflect its use of small local views. Constructing these views requires neighborhood selection and data indexing, which contribute to solution time but are excluded from the neural FLOPs count.

On ATSP-1K, AR-GCN requires 138.7 K GFLOPs per solution, 33.2% fewer than GOAL's 207.6 K. The main results also show a 26.8% reduction in optimality gap (2.24% to 1.64%) and a 20.6% reduction in solution time (3.64m to 2.89m) compared with GOAL (Table 3). AR-GCN achieves these results with only 0.628M learnable parameters.

F Complete Results on TSPLIB and CVRPLIB

The complete results on TSPLIB and CVRPLIB are reported in Tables 18 to 24, where the per-instance optimality gaps (%) of all neural methods are listed in detail. Overall, AR-GCN excels on STSPs and ATSPs, while remaining highly competitive on SCVRPs, where further improvements are still possible. Moreover, GOAL [17] delivers more consistent and generally better results on SCVRPs, but requires over 3 times as many parameters as AR-GCN and higher GPU memory demands.

G Limitations and Future Work

The current Min-Max Clamp still requires further validation and improvement of its robustness across a broader range of AVRP distributions, particularly those with pronounced outliers. In future work, it is also necessary to investigate low-loss graph sparsification strategies for large-scale AVRPs to reduce the computational cost. Moreover, AR-GCN is expected to be extended to a wider spectrum of combinatorial optimization problems, such as Open Vehicle Routing Problems (OVRP), Job Shop Scheduling Problems (JSSP), and Maximum Independent Set Problems (MIS). Furthermore, hybrid architectures that combine Gating Mechanism with Attention Mechanism are promising, as they can leverage the complementary strengths of both paradigms to achieve more efficient performance.

Table 17: Neural forward computation per instance (GFLOPs) for one solution. OOM denotes out-of-memory on the 24 GB profiling device, and — denotes an unsupported setting or one absent from the main comparison.

Task	**STSP**				**SCVRP**			
Scale	100	1K	5K	10K	100	1K	5K	10K
POMO	0.3	6.6	116.0	439.1	0.3	6.8	118.9	450.4
Omni-VRP	0.3	6.6	116.0	439.1	0.3	6.8	119.2	454.2
LEHD	13.4	2.3 K	167.7 K	1222.9 K	13.7	2.3 K	167.8 K	1223.3 K
ELG	0.3	6.8	116.9	441.0	0.3	7.0	119.7	452.3
INViT	4.3	49.6	250.6	502.0	11.0	110.5	551.7	1.1 K
SIGD	20.1	3.5 K	251.9 K	1835.7 K	35.4	1.8 K	10.0 K	20.2 K
ICAM	0.5	11.7	202.6	765.7	0.5	12.0	209.5	793.8
GREAT	13.5	1.4 K	OOM	OOM	18.8	1.9 K	OOM	OOM
GOAL	—	—	—	—	239.4	6.1 K	32.1 K	64.6 K
AR-GCN$_{50}$	78.0	1.1 K	5.8 K	11.6 K	79.2	1.1 K	5.8 K	11.6 K
AR-GCN$_{100}$	148.3	4.1 K	21.6 K	43.5 K	152.7	4.1 K	21.6 K	43.5 K

Task	**ATSP**				**ACVRP**			
Scale	50	100	500	1K	50	100	500	1K
MatNet	0.6	1.4	—	—	—	—	—	—
ICAM	0.2	0.5	4.1	11.8	—	—	—	—
GREAT	4.6	18.5	463.2	1.9 K	4.8	18.8	465.3	1.9 K
MatPOENet	1.2	2.5	18.9	53.4	—	—	—	—
GOAL	32.7	232.6	26.3 K	207.6 K	—	—	—	—
RADAR	0.3	0.7	6.9	21.6	0.3	0.8	7.0	21.9
RRNCO	2.3	8.8	208.0	826.1	2.4	9.0	208.9	828.1
AR-GCN	19.9	148.3	17.5 K	138.7 K	21.1	152.7	17.6 K	139.1 K

Table 18: Optimality gaps (%) for each ATSP instance in TSPLIB. (ERR indicates runtime errors.)

Instance	ICAM	GREAT	MatPOENet	GOAL	RADAR	RRNCO	AR-GCN
br17	ERR	**0.00**	12.82	120.51	2.56	**0.00**	**0.00**
ftv33	ERR	12.36	32.12	13.30	18.66	12.60	**3.34**
ftv35	ERR	8.55	32.45	3.46	31.43	15.82	**0.00**
ftv38	ERR	12.55	36.99	5.69	16.01	11.24	**0.00**
p43	ERR	2.62	10.48	20.28	8.24	12.08	**1.05**
ftv44	ERR	6.63	38.62	5.95	18.97	28.27	**2.73**
ftv47	ERR	12.50	48.14	5.74	**1.24**	13.12	2.98
ry48p	ERR	10.75	24.95	10.98	60.46	12.50	**6.30**
ft53	37.68	10.59	68.23	21.49	55.35	37.54	**4.87**
ftv55	55.22	12.94	49.13	13.25	15.17	17.91	**2.11**
ftv64	67.92	19.25	51.33	6.53	2.07	25.01	**0.76**
ft70	43.86	3.33	45.86	5.11	4.63	11.78	**2.17**
ftv70	61.23	11.69	53.95	5.85	4.92	20.56	**4.00**
ftv90	38.32	36.23	62.19	10.58	34.58	42.88	**6.27**
kro124p	47.64	19.95	25.33	12.80	62.69	15.52	**8.70**
ftv100	40.94	40.49	64.88	9.23	43.01	49.50	**4.47**
ftv110	50.36	37.69	70.94	10.42	42.75	45.81	**3.78**
ftv120	50.55	41.18	80.10	11.96	35.04	48.52	**8.86**
ftv130	44.00	34.16	68.53	**4.81**	45.99	37.71	8.89
ftv140	60.17	39.46	74.46	**5.91**	50.04	54.92	6.78
ftv150	59.94	32.59	74.22	9.35	60.09	61.24	**5.48**
ftv160	67.31	34.10	77.00	15.24	69.88	56.95	**6.78**
ftv170	61.89	49.15	98.69	**4.17**	56.04	75.43	12.96
rbg323	113.05	346.68	68.55	82.96	77.00	28.28	**14.03**
rbg358	113.84	506.10	84.01	168.53	72.48	**45.06**	52.54
rbg403	40.37	217.65	83.65	59.88	64.18	34.65	**24.83**
rbg443	41.43	216.18	85.44	39.67	78.01	38.60	**24.56**

Table 19: Optimality gaps (%) for each STSP instance in TSPLIB. (OOM indicates out-of-memory issues.)

Instance	POMO	Omni-VRP	LEHD	ELG	INViT	SIGD	ICAM	GREAT	AR-GCN$_{50}$	AR-GCN$_{100}$
eil51	1.40	1.74	1.64	0.47	3.52	0.67	1.51	4.87	**0.00**	**0.00**
berlin52	0.04	7.75	0.03	0.01	2.32	0.03	7.64	10.03	**0.00**	**0.00**
st70	0.33	2.63	0.33	0.15	2.96	0.31	0.51	2.92	0.59	**0.00**
eil76	1.24	6.32	2.54	1.49	4.65	1.24	1.90	4.82	**0.37**	**0.37**
pr76	0.22	2.14	0.22	0.83	1.43	**0.00**	3.46	2.34	0.82	0.21
rat99	6.45	2.65	1.10	5.04	2.56	0.68	15.97	13.65	**0.17**	**0.17**
kroA100	1.52	1.11	0.12	1.99	1.26	0.02	4.66	6.21	**0.00**	**0.00**
kroB100	1.10	4.03	0.26	2.13	2.28	0.01	5.89	6.83	0.23	**0.00**
kroC100	1.49	1.32	0.32	0.10	1.26	**0.01**	2.45	11.26	0.02	0.02
kroD100	3.79	3.32	0.38	2.93	2.90	**0.00**	4.12	9.97	**0.00**	0.38
kroE100	2.12	2.32	0.43	2.01	2.28	**0.05**	1.99	4.62	0.17	0.15
rd100	0.01	1.36	0.01	0.58	2.93	**0.00**	0.14	3.81	**0.00**	**0.00**
eil101	2.06	3.70	2.31	1.43	4.61	1.78	2.67	6.06	**0.00**	**0.00**
lin105	2.57	6.42	0.34	0.58	3.00	0.03	9.59	8.52	**0.00**	**0.00**
pr107	3.90	1.14	11.82	3.32	**0.90**	14.24	3.92	8.60	1.18	11.27
pr124	0.97	1.55	1.11	0.75	2.35	1.42	1.54	2.61	0.86	**0.50**
bier127	10.22	5.59	5.63	8.86	12.46	**0.20**	8.40	14.63	1.65	0.41
ch130	0.85	3.94	0.55	1.44	2.93	**0.25**	0.48	5.27	0.51	0.54
pr136	1.01	1.59	0.45	1.59	2.96	**0.00**	0.69	23.37	0.43	0.23
pr144	0.85	2.21	3.59	0.94	3.88	**0.03**	0.75	4.07	4.36	1.07
ch150	0.73	1.54	0.58	1.32	2.82	0.29	0.44	5.05	**0.00**	0.43
kroA150	1.72	2.78	1.40	2.73	4.00	**0.00**	4.02	8.31	0.54	**0.00**
kroB150	3.06	2.12	0.76	1.84	4.58	**0.00**	2.82	12.02	0.15	0.01
pr152	2.67	1.58	12.14	**1.18**	7.72	8.89	5.21	13.59	8.31	9.66
u159	1.75	1.34	1.45	1.55	0.85	**0.00**	1.53	6.97	**0.00**	**0.00**
rat195	13.46	7.33	1.42	4.00	3.57	**0.70**	11.38	12.01	2.80	0.77
d198	40.38	5.86	9.23	9.84	15.14	8.65	24.55	39.58	6.57	**2.00**
kroA200	3.17	2.46	0.64	2.76	2.88	0.49	4.77	9.09	1.38	**0.22**
kroB200	4.97	3.63	0.16	3.60	3.42	**0.08**	4.81	11.64	0.23	0.21
ts225	10.01	2.77	**0.28**	1.35	3.56	0.36	5.34	23.44	2.20	2.72
tsp225	8.83	5.31	0.67	5.41	2.45	**0.13**	4.35	10.21	0.54	0.61
pr226	3.93	2.35	0.87	2.04	6.82	**0.39**	1.69	7.80	10.18	0.96
gil262	4.53	3.93	1.60	3.28	3.70	**0.51**	0.80	13.26	1.05	0.67
pr264	17.55	3.56	5.17	3.85	6.51	3.08	4.63	17.99	**0.67**	3.72
a280	14.41	6.92	3.02	7.02	7.13	1.14	7.37	19.92	2.64	**0.54**
pr299	15.09	5.59	2.91	5.88	6.65	**0.47**	13.97	29.18	2.95	1.18
lin318	11.09	4.84	1.41	6.06	5.64	**0.11**	2.37	13.34	1.19	1.53
rd400	16.22	6.56	1.00	8.97	4.78	**0.46**	1.04	16.79	1.55	0.80
fl417	17.38	9.23	**4.78**	6.97	11.66	43.46	28.25	17.44	12.12	14.60
pr439	26.77	7.52	3.37	7.63	11.69	**1.21**	6.55	24.57	1.73	4.57
pcb442	19.32	9.72	3.11	10.72	3.03	**0.36**	3.09	17.44	3.92	2.70
d493	51.10	12.02	9.49	30.01	4.50	**4.18**	19.86	33.82	6.26	4.89
u574	31.60	11.80	2.73	13.39	8.81	**0.86**	10.97	23.46	2.77	2.91
rat575	25.66	15.05	3.02	11.09	5.73	**1.19**	13.90	25.22	2.07	2.50
p654	28.87	14.04	10.24	10.47	13.98	**7.02**	28.96	22.22	13.78	7.67
d657	33.72	13.33	8.05	14.64	10.79	**1.98**	13.62	22.50	7.32	3.50
u724	27.91	15.43	3.27	12.90	5.97	**0.94**	13.09	27.03	2.95	1.31
rat783	33.07	20.35	4.27	14.08	5.60	1.72	15.57	25.05	**1.65**	3.71
pr1002	40.47	19.79	4.44	12.95	10.37	**2.52**	12.37	28.08	4.42	6.96
u1060	47.00	27.49	8.92	15.13	8.84	**4.35**	20.11	38.53	8.86	4.81
vm1084	41.97	22.64	5.98	16.90	7.16	7.00	14.58	35.67	3.42	**1.96**
pcb1173	45.12	28.43	6.34	18.57	7.21	**2.84**	15.02	25.62	3.82	3.69
d1291	66.76	33.35	14.13	12.69	12.33	**7.11**	13.22	41.83	11.05	8.83
rl1304	56.18	31.22	7.75	19.04	9.02	5.51	14.96	37.83	5.81	**2.93**
rl1323	62.81	31.48	9.26	15.16	9.52	**3.69**	12.65	37.11	10.26	4.59
nrw1379	38.61	24.78	9.91	13.71	6.27	**2.65**	12.76	29.23	2.68	5.30
fl1400	43.33	22.73	18.80	15.76	**9.55**	44.43	23.23	35.27	10.52	9.86
u1432	33.52	22.25	3.34	11.63	3.78	**1.32**	11.84	26.53	1.81	3.03
fl1577	61.80	33.70	17.62	15.53	10.27	27.51	20.91	34.34	9.41	**6.89**
d1655	60.94	34.55	13.89	15.78	13.04	13.11	15.35	37.43	**5.66**	8.09
vm1748	58.72	30.47	14.81	20.85	9.35	5.89	16.95	35.67	4.37	**3.59**
u1817	61.02	40.73	10.28	14.91	8.68	**5.96**	13.96	46.44	8.98	6.40
rl1889	62.39	39.68	10.64	15.54	9.89	8.57	13.53	44.36	7.94	**5.38**
d2103	65.10	36.20	14.57	11.34	9.97	12.72	12.90	31.48	6.05	**4.08**
u2152	62.65	42.13	11.61	17.70	7.63	6.85	16.38	51.27	**4.86**	6.21
u2319	23.84	19.66	2.53	5.28	0.98	0.70	6.83	OOM	**0.22**	0.32
pr2392	59.29	41.20	10.98	21.53	8.86	**5.53**	14.57	OOM	8.06	6.70
pcb3038	OOM	40.80	13.04	OOM	7.02	7.52	17.31	OOM	**5.83**	6.20
fl3795	OOM	54.40	13.55	OOM	10.75	61.41	22.09	OOM	**7.03**	10.34
fnl4461	OOM	45.15	18.79	OOM	6.93	19.13	18.44	OOM	**4.62**	5.22
rl5915	OOM	OOM	22.34	OOM	10.09	33.78	20.21	OOM	**7.74**	8.53
rl5934	OOM	OOM	35.47	OOM	11.14	46.57	19.45	OOM	9.38	**8.54**

Table 20: Optimality gaps (%) for each STSP instance in TSPLIB (continued from Table 19).

Instance	POMO	Omni-VRP	LEHD	ELG	INViT	SIGD	ICAM	GREAT	AR-GCN$_{50}$	AR-GCN$_{100}$
rl11849	OOM	OOM	41.69	OOM	9.69	OOM	21.38	OOM	**7.89**	8.31
usa13509	OOM	OOM	55.80	OOM	10.32	OOM	OOM	OOM	**8.91**	10.32
brd14051	OOM	OOM	OOM	OOM	8.60	OOM	OOM	OOM	6.81	**6.76**
d15112	OOM	OOM	OOM	OOM	7.84	OOM	OOM	OOM	**5.29**	6.74
d18512	OOM	OOM	OOM	OOM	7.87	OOM	OOM	OOM	**5.22**	7.57

Table 21: Optimality gaps (%) for each SCVRP instance in Set X of CVRPLIB.

Instance	POMO	Omni-VRP	LEHD	ELG	INViT	SIGD	ICAM	GREAT	GOAL	AR-GCN$_{50}$	AR-GCN$_{100}$
X-n101-k25	6.64	8.48	13.97	6.84	6.49	13.08	7.35	8.19	11.88	**5.86**	6.25
X-n106-k14	3.07	3.68	3.75	4.62	5.01	1.90	4.17	3.30	**1.51**	3.58	3.68
X-n110-k13	3.25	5.43	1.88	5.28	5.53	1.49	2.11	5.83	**1.39**	3.02	2.08
X-n115-k10	15.69	12.22	9.43	10.43	8.90	24.68	9.37	**6.06**	12.12	6.65	6.28
X-n120-k6	15.34	10.94	3.89	7.93	7.62	**1.48**	3.75	7.74	1.58	2.44	2.30
X-n125-k30	**5.19**	8.41	17.88	7.89	8.42	44.40	6.89	**5.19**	7.72	9.97	11.66
X-n129-k18	3.51	4.66	4.01	3.49	7.63	3.63	1.87	4.39	**1.63**	4.63	3.72
X-n134-k13	5.55	6.49	8.94	7.55	4.39	5.09	10.25	8.98	**3.54**	4.16	4.17
X-n139-k10	6.39	3.58	3.08	5.87	4.36	2.29	3.87	11.98	**1.64**	4.41	3.75
X-n143-k7	6.47	8.18	14.00	8.46	8.33	4.69	3.92	8.77	**3.46**	7.06	3.94
X-n148-k46	11.32	8.88	38.98	**5.08**	11.69	59.15	6.32	45.68	11.09	11.86	6.46
X-n153-k22	16.40	**10.72**	28.93	13.26	13.58	146.50	13.68	17.67	16.38	12.43	19.21
X-n157-k13	9.46	2.17	4.72	6.59	12.79	3.90	9.29	4.61	7.47	**1.89**	4.60
X-n162-k11	7.99	6.77	3.47	4.64	4.75	3.28	5.03	7.95	2.67	**2.17**	2.84
X-n167-k10	5.45	6.66	5.03	5.24	6.47	2.53	3.72	4.74	**1.16**	2.94	4.37
X-n172-k51	14.56	10.41	33.28	7.87	7.78	89.78	**5.46**	116.95	8.27	9.24	9.08
X-n176-k26	15.40	10.90	27.00	11.89	13.24	100.82	13.16	8.86	17.13	9.87	**7.10**
X-n181-k23	6.02	2.52	1.43	3.32	14.92	1.38	3.54	2.92	**0.98**	1.27	1.19
X-n186-k15	8.91	4.24	3.64	3.78	7.49	1.89	3.84	7.93	**1.55**	3.57	6.50
X-n190-k8	9.69	5.61	5.41	9.74	7.47	**1.97**	5.14	10.07	5.26	5.64	4.72
X-n195-k51	16.07	11.70	15.47	7.65	8.99	59.32	**6.19**	18.93	9.37	8.23	6.34
X-n200-k36	6.62	6.30	10.25	7.90	5.36	7.51	**4.76**	5.34	5.15	5.17	6.83
X-n204-k19	9.44	4.20	2.68	5.39	10.32	**2.06**	3.74	6.29	2.13	5.11	5.42
X-n209-k16	7.48	4.21	3.88	4.57	7.42	3.88	2.32	6.74	**1.71**	2.60	2.97
X-n214-k11	11.60	5.22	7.21	6.13	12.45	**2.50**	4.39	11.88	2.64	5.33	4.71
X-n219-k73	5.41	2.54	15.51	3.11	47.64	12.19	**1.18**	50.54	**1.18**	3.22	16.55
X-n223-k34	10.02	6.28	5.80	5.20	9.33	5.33	**3.35**	5.21	4.94	4.67	5.27
X-n228-k23	16.84	9.93	14.40	13.46	14.62	154.00	9.95	9.92	**7.23**	11.95	12.19
X-n233-k16	10.33	7.26	5.63	8.42	11.95	5.00	**3.74**	9.99	4.00	6.49	5.27
X-n237-k14	19.01	4.62	4.05	5.89	10.68	4.13	5.73	7.58	**1.56**	3.51	3.31
X-n242-k48	9.33	5.90	6.02	3.99	10.45	7.01	4.19	**3.15**	6.81	4.20	6.12
X-n247-k50	15.17	11.80	44.92	**11.13**	14.29	92.50	14.04	65.47	16.98	13.96	14.10
X-n251-k28	8.83	3.80	2.37	5.44	8.27	4.51	2.83	6.29	2.68	**1.88**	2.46
X-n256-k16	20.36	5.64	7.29	6.35	7.52	4.65	6.77	9.55	**1.55**	3.90	3.22
X-n261-k13	11.17	5.22	8.82	6.65	7.77	3.89	3.10	10.62	**1.89**	4.62	5.92
X-n266-k58	13.67	8.42	6.38	5.69	8.07	5.60	5.68	5.20	5.23	**4.69**	5.25
X-n270-k35	18.88	7.52	6.54	5.03	8.16	4.06	4.05	6.06	5.59	5.55	**3.86**
X-n275-k28	16.31	5.74	6.58	7.76	14.00	2.60	4.59	6.41	**1.17**	3.82	4.22
X-n280-k17	14.14	6.02	9.64	9.80	11.26	14.32	**5.31**	9.69	9.99	9.54	8.14
X-n284-k15	14.73	5.66	4.99	8.32	9.60	5.64	7.24	12.31	**3.54**	6.46	6.34
X-n289-k60	12.52	7.25	9.90	6.51	7.97	12.68	5.34	4.64	8.46	**4.53**	6.53
X-n294-k50	17.16	12.29	9.68	8.63	10.60	8.28	**4.08**	6.32	7.18	6.91	7.06
X-n298-k31	15.81	8.13	7.65	8.69	10.37	4.48	2.82	8.61	**1.50**	5.85	3.79
X-n303-k21	15.93	4.67	**2.83**	7.33	9.63	4.49	4.00	11.62	3.82	4.59	4.59
X-n308-k13	18.98	10.11	**4.49**	10.90	12.33	7.81	7.20	12.92	4.76	5.04	6.05
X-n313-k71	12.36	8.45	14.06	5.98	7.73	16.91	**4.56**	5.69	7.34	6.34	8.02
X-n317-k53	21.99	3.47	3.88	6.96	21.53	3.13	2.62	3.53	1.67	**1.13**	4.23
X-n322-k28	20.17	7.64	3.89	8.64	11.63	4.67	3.73	12.20	**2.81**	4.16	4.21
X-n327-k20	21.21	6.96	10.74	8.07	8.99	4.48	5.52	11.25	**2.32**	3.58	3.82
X-n331-k15	42.40	4.94	3.36	6.13	10.08	2.05	2.43	12.00	**1.49**	5.24	4.34
X-n336-k84	13.03	7.26	18.78	5.90	7.61	36.70	5.91	**5.05**	8.43	7.56	12.03
X-n344-k43	19.28	9.59	3.93	6.37	10.76	5.55	3.78	7.27	**3.30**	3.85	5.11
X-n351-k40	23.30	8.75	7.48	8.26	10.18	7.62	9.76	8.80	**3.63**	4.53	4.46
X-n359-k29	11.32	4.78	2.79	5.01	8.17	3.45	1.90	8.58	**1.72**	3.76	3.86
X-n367-k17	22.68	6.98	9.44	9.26	9.08	14.26	11.29	13.82	7.85	**6.04**	6.21
X-n376-k94	11.34	3.05	5.45	5.67	33.07	3.52	1.25	2.75	**1.16**	1.82	23.86
X-n384-k52	18.51	8.23	7.79	6.33	6.40	4.66	3.64	6.90	**3.53**	6.02	3.58
X-n393-k38	25.07	8.17	4.47	7.35	10.21	4.38	3.39	10.42	**1.90**	5.56	5.64
X-n401-k29	11.94	4.54	5.55	4.87	7.09	5.19	**3.00**	8.48	3.93	6.14	4.58
X-n411-k19	38.76	12.31	13.49	14.46	15.72	20.65	12.66	23.60	9.23	**7.77**	11.55
X-n420-k130	20.84	11.45	61.64	7.69	15.65	57.73	**5.30**	110.80	10.61	11.49	10.33
X-n429-k61	28.79	10.21	6.25	7.89	7.66	**3.78**	4.08	6.99	4.31	5.55	4.40
X-n439-k37	33.91	7.70	2.00	6.05	16.94	4.83	8.66	10.77	**1.44**	2.76	1.52
X-n449-k29	27.68	6.19	7.18	8.09	8.50	4.19	**3.20**	10.69	3.32	5.43	4.86
X-n459-k26	40.93	7.33	8.66	11.59	12.46	6.30	7.10	20.22	**2.93**	6.41	5.69
X-n469-k138	15.27	8.75	28.32	8.07	6.89	15.17	6.91	7.26	**5.87**	7.35	9.88
X-n480-k70	19.61	7.60	3.06	7.11	7.90	6.71	3.90	7.45	**2.46**	4.05	4.11
X-n491-k59	27.15	8.85	5.07	7.29	7.73	6.49	**3.22**	8.35	6.28	4.93	5.04
X-n502-k39	30.26	3.13	3.31	7.62	11.67	2.34	3.75	5.07	**1.42**	1.44	1.53
X-n513-k21	52.81	11.14	4.97	11.69	13.88	4.85	5.53	24.99	**4.66**	6.07	5.65

Table 22: Optimality gaps (%) for each SCVRP instance in Set X of CVRPLIB (continued from Table 21).

Instance	POMO	Omni-VRP	LEHD	ELG	INViT	SIGD	ICAM	GREAT	GOAL	AR-GCN$_{50}$	AR-GCN$_{100}$
X-n524-k153	19.87	13.54	78.66	13.40	13.43	83.89	**11.99**	210.04	19.19	17.54	28.30
X-n536-k96	26.41	8.86	13.92	8.42	7.54	16.43	6.63	11.22	**3.73**	6.45	7.58
X-n548-k50	12.52	5.12	3.32	5.35	13.92	1.63	2.01	9.12	**0.78**	2.11	1.51
X-n561-k42	72.01	8.84	5.86	9.67	13.71	4.98	3.53	14.24	**3.03**	5.37	4.71
X-n573-k30	24.57	5.68	11.35	20.21	7.55	20.64	7.78	64.05	7.35	6.38	**4.82**
X-n586-k159	15.92	9.46	17.12	7.57	10.25	13.65	5.94	7.68	**5.43**	6.05	8.72
X-n599-k92	31.57	8.95	8.92	7.00	6.66	5.09	4.93	7.81	**4.35**	5.70	4.48
X-n613-k62	69.80	11.89	6.89	10.03	12.26	6.52	4.11	15.07	**3.67**	6.82	5.31
X-n627-k43	32.94	6.79	5.49	9.93	7.87	6.13	7.00	11.11	**2.88**	4.76	4.78
X-n641-k35	21.03	6.87	5.65	7.89	8.87	4.07	4.12	14.18	**3.18**	4.66	3.94
X-n655-k131	26.84	3.81	5.24	5.77	26.04	3.62	3.02	7.95	**1.01**	1.62	13.63
X-n670-k130	23.21	15.26	98.95	17.31	15.91	106.28	**12.43**	228.12	21.53	19.64	26.98
X-n685-k75	78.75	10.71	8.34	9.73	11.28	8.14	**4.30**	14.34	5.00	6.41	6.57
X-n701-k44	46.36	5.88	4.11	6.56	8.30	4.49	**2.31**	12.45	2.83	3.07	3.17
X-n716-k35	22.91	6.55	8.80	10.33	8.48	5.72	6.00	16.13	**5.13**	7.89	6.60
X-n733-k159	55.21	12.96	14.72	6.67	12.17	16.71	**4.83**	11.48	5.86	5.40	7.65
X-n749-k98	42.09	9.69	8.97	8.06	7.94	8.27	4.72	12.34	**4.58**	5.08	4.77
X-n766-k71	46.60	9.78	15.49	9.55	12.46	76.70	7.73	14.49	13.05	9.18	**7.12**
X-n783-k48	106.25	9.23	5.64	9.60	8.73	5.53	3.74	18.09	**2.58**	4.81	3.51
X-n801-k40	57.72	7.84	3.48	8.94	12.76	4.64	5.09	14.24	**1.45**	3.71	4.67
X-n819-k171	55.20	9.81	12.22	8.76	7.77	12.22	6.22	9.92	**5.20**	5.58	6.64
X-n837-k142	23.42	8.32	6.68	6.74	8.38	6.15	3.89	8.47	3.31	**3.25**	4.20
X-n856-k95	89.05	9.61	2.97	8.83	17.35	3.65	5.15	13.86	**1.53**	2.72	2.24
X-n876-k59	71.77	5.30	7.30	6.73	7.25	5.90	4.19	16.45	**3.30**	4.77	3.76
X-n895-k37	53.37	10.68	8.70	12.84	9.15	6.18	4.80	21.02	**2.90**	6.15	6.39
X-n916-k207	23.72	8.21	15.36	6.95	8.89	8.46	5.04	8.88	4.54	**4.21**	5.64
X-n936-k151	65.86	20.56	139.42	17.73	21.35	142.55	**17.36**	322.22	19.65	24.50	21.98
X-n957-k87	89.79	8.95	3.85	9.53	15.46	4.11	3.66	13.80	**1.36**	2.61	2.51
X-n979-k58	60.42	6.50	19.79	6.33	7.72	6.63	4.90	14.37	**3.51**	5.40	4.35
X-n1001-k43	26.72	11.08	7.57	9.87	10.03	6.63	**3.49**	20.23	4.19	4.77	4.45

Table 23: Optimality gaps (%) for each SCVRP instance in Set XL of CVRPLIB.

Instance	POMO	Omni-VRP	LEHD	ELG	INViT	SIGD	ICAM	GREAT	GOAL	AR-GCN$_{50}$	AR-GCN$_{100}$
XL-n1048-k237	29.77	12.94	72.75	12.83	14.67	60.80	**11.33**	127.35	19.79	16.24	31.50
XL-n1094-k157	101.46	7.86	4.85	11.06	20.52	5.85	5.55	10.77	1.46	2.06	**1.38**
XL-n1141-k112	146.50	14.36	7.57	11.04	9.00	6.58	4.65	17.16	**2.89**	4.96	4.60
XL-n1188-k96	70.33	10.05	8.64	10.62	8.30	7.62	4.03	34.54	**3.39**	7.25	4.80
XL-n1234-k55	47.88	10.95	6.06	10.03	10.11	5.56	3.71	19.30	**3.28**	4.95	4.38
XL-n1281-k29	120.51	20.36	14.50	20.89	16.87	12.05	10.66	51.31	**6.33**	7.71	8.07
XL-n1328-k19	144.87	25.71	13.71	21.21	17.02	12.98	12.51	48.77	**9.06**	9.96	10.17
XL-n1374-k278	111.52	13.45	25.37	10.01	8.67	17.42	6.82	14.47	**5.41**	7.50	9.61
XL-n1421-k232	19.61	5.77	42.35	6.77	6.01	10.38	**3.45**	95.72	7.11	5.96	4.52
XL-n1468-k151	62.84	8.68	6.71	7.20	7.09	4.52	3.57	19.26	**3.31**	3.97	3.58
XL-n1514-k106	322.28	14.53	7.76	13.04	12.10	8.45	5.89	OOM	**2.82**	4.98	4.61
XL-n1561-k75	45.08	7.13	5.68	9.76	11.59	6.56	5.75	OOM	**2.82**	3.04	2.95
XL-n1608-k39	689.30	29.63	12.30	30.54	18.48	16.89	12.20	OOM	**5.97**	10.18	9.09
XL-n1654-k11	142.42	48.94	20.68	29.86	22.61	12.39	20.00	OOM	**10.33**	12.25	11.95
XL-n1701-k562	21.02	11.74	96.71	12.08	**8.91**	20.99	9.12	OOM	9.96	9.80	26.02
XL-n1748-k271	30.44	12.88	15.44	12.40	8.65	15.10	7.74	OOM	**4.96**	7.09	8.89
XL-n1794-k163	164.59	15.53	4.85	11.88	16.70	4.63	5.47	OOM	**1.24**	1.87	2.07
XL-n1841-k126	88.45	9.82	28.39	8.85	10.83	43.98	**4.69**	OOM	8.94	6.10	5.77
XL-n1888-k82	186.71	12.29	6.67	9.75	9.24	6.76	4.86	OOM	**2.59**	4.57	4.13
XL-n1934-k46	198.58	22.65	13.78	18.48	17.23	15.97	10.62	OOM	**6.00**	12.50	9.56
XL-n1981-k13	260.08	52.06	22.23	27.27	28.71	17.97	24.08	OOM	17.64	**13.67**	14.29
XL-n2028-k617	25.97	13.79	74.06	8.14	7.63	15.66	7.27	OOM	7.74	**6.91**	21.04
XL-n2074-k264	44.78	8.38	13.51	6.91	6.43	7.16	**3.29**	OOM	3.54	4.71	3.77
XL-n2121-k186	64.19	8.96	6.94	7.56	8.33	6.90	2.89	OOM	2.77	2.97	**2.31**
XL-n2168-k138	502.31	18.68	8.39	15.87	13.27	9.96	6.55	OOM	**4.14**	8.02	7.03
XL-n2214-k131	73.64	8.50	6.09	8.18	12.51	8.24	5.69	OOM	2.97	**2.71**	2.91
XL-n2261-k54	62.35	18.13	8.41	13.51	13.48	11.25	6.43	OOM	5.96	6.97	**5.83**
XL-n2307-k34	1255.44	49.68	13.42	34.46	25.63	17.81	14.91	OOM	**11.94**	13.39	12.15
XL-n2354-k631	18.25	8.33	69.25	8.13	10.31	12.47	6.08	OOM	6.97	**4.14**	17.17
XL-n2401-k408	119.66	11.78	24.06	9.45	7.76	11.93	5.81	OOM	**5.08**	7.40	7.18
XL-n2447-k290	174.54	18.02	94.00	19.34	17.78	125.72	**10.44**	OOM	17.00	18.45	15.53
XL-n2494-k194	23.14	5.05	16.39	13.86	5.95	6.31	4.42	OOM	**3.36**	6.98	4.30
XL-n2541-k121	79.88	15.68	7.09	11.94	13.30	8.78	6.50	OOM	**1.57**	3.16	2.98
XL-n2587-k66	840.90	33.58	11.78	21.16	22.11	14.18	13.46	OOM	**6.34**	9.37	8.88
XL-n2634-k17	321.09	58.46	27.02	36.88	31.25	32.44	32.15	OOM	18.76	18.29	**15.61**
XL-n2681-k540	27.23	7.88	76.54	6.38	5.53	11.50	**2.90**	OOM	6.69	6.00	4.99
XL-n2727-k546	152.08	9.73	18.96	13.70	26.40	8.60	3.73	OOM	**0.85**	1.60	16.54
XL-n2774-k286	36.52	8.32	18.82	7.85	6.46	5.09	4.03	OOM	2.87	3.72	**2.84**
XL-n2821-k208	352.22	18.56	18.08	15.54	13.65	69.83	**7.37**	OOM	10.72	9.53	8.23
XL-n2867-k120	50.67	13.60	11.45	10.99	11.12	8.22	7.41	OOM	4.41	4.26	**3.73**
XL-n2914-k95	769.46	26.19	12.67	19.16	16.78	14.61	9.46	OOM	**5.84**	10.32	8.97
XL-n2961-k55	66.00	25.57	11.86	17.85	14.80	11.26	9.14	OOM	**7.00**	9.58	8.23
XL-n3007-k658	154.90	27.35	87.10	8.30	10.29	15.78	**5.68**	OOM	5.76	5.70	7.05
XL-n3054-k461	57.38	8.25	10.43	5.71	6.07	4.97	4.79	OOM	**3.28**	4.27	3.76
XL-n3101-k311	82.20	15.12	207.33	14.03	13.33	83.14	**6.89**	OOM	18.39	11.44	8.30
XL-n3147-k232	321.54	16.30	8.20	10.06	9.08	7.85	4.43	OOM	**2.83**	3.32	3.14
XL-n3194-k161	689.17	36.95	10.96	17.47	12.65	11.95	7.86	OOM	**3.97**	6.90	6.44

Table 24: Optimality gaps (%) for each SCVRP instance in Set XL of CVRPLIB. (continued from Table 23).

Instance	POMO	Omni-VRP	LEHD	ELG	INViT	SIGD	ICAM	GREAT	GOAL	AR-GCN$_{50}$	AR-GCN$_{100}$
XL-n3241-k115	31.58	10.25	9.33	25.51	9.78	11.44	6.54	OOM	**4.66**	10.61	**4.66**
XL-n3287-k30	546.59	39.16	46.19	32.16	23.49	30.15	24.40	OOM	**9.70**	12.01	12.58
XL-n3334-k934	12.04	9.46	70.80	4.77	9.15	29.60	**3.63**	OOM	7.12	8.46	6.70
XL-n3408-k524	34.20	13.27	33.68	8.51	7.28	7.66	**4.71**	OOM	5.53	7.97	6.30
XL-n3484-k436	41.45	4.87	14.75	5.56	17.64	2.73	2.08	OOM	1.09	**0.95**	4.14
XL-n3561-k229	242.19	26.70	10.04	14.37	11.36	12.68	5.15	OOM	**3.83**	6.88	6.10
XL-n3640-k211	518.59	38.47	9.33	14.94	12.39	10.75	6.52	OOM	**2.68**	4.98	4.33
XL-n3721-k77	509.42	25.88	16.23	17.45	14.94	20.60	8.98	OOM	**7.12**	7.87	7.19
XL-n3804-k29	1196.24	79.78	24.66	33.86	30.21	20.16	26.33	OOM	**13.01**	15.44	14.40
XL-n3888-k1010	23.84	10.14	68.11	12.21	9.39	29.15	**8.69**	OOM	18.25	12.24	15.99
XL-n3975-k687	197.14	23.99	23.45	8.04	10.19	15.51	5.02	OOM	4.54	**3.87**	4.48
XL-n4063-k347	180.55	14.11	15.88	8.33	6.07	6.65	6.18	OOM	3.82	5.91	**3.56**
XL-n4153-k291	36.44	9.41	16.81	7.83	7.36	9.94	4.46	OOM	**3.60**	8.08	4.90
XL-n4245-k203	117.36	20.97	9.80	10.75	12.87	9.33	7.10	OOM	**2.10**	2.90	2.63
XL-n4340-k148	84.91	20.19	10.18	11.76	9.69	10.42	8.08	OOM	**3.71**	4.62	4.28
XL-n4436-k48	431.67	42.19	30.93	29.54	21.61	26.58	20.36	OOM	**7.91**	11.91	10.43
XL-n4535-k1134	76.94	5.09	54.56	6.91	33.96	4.91	1.27	OOM	3.10	**0.56**	31.91
XL-n4635-k790	195.08	33.15	20.30	10.18	7.68	19.10	14.48	OOM	5.57	**4.99**	5.28
XL-n4738-k487	133.55	16.63	13.95	7.45	6.55	6.63	7.92	OOM	3.83	6.27	**3.56**
XL-n4844-k321	271.19	23.37	23.36	OOM	11.54	54.16	7.21	OOM	9.72	6.83	**6.18**
XL-n4951-k203	233.71	18.97	9.42	OOM	9.52	9.55	6.28	OOM	**3.30**	4.11	3.68
XL-n5061-k184	155.12	18.09	16.66	OOM	12.75	11.99	9.58	OOM	**3.74**	4.64	4.08
XL-n5174-k55	1272.57	84.42	32.62	OOM	24.82	29.36	23.63	OOM	14.51	14.34	**12.31**
XL-n5288-k1246	42.81	15.09	47.18	OOM	6.24	5.95	6.87	OOM	6.12	**3.44**	8.93
XL-n5406-k783	49.51	14.06	20.41	OOM	7.20	7.25	2.65	OOM	2.91	2.63	**2.55**
XL-n5526-k553	138.77	OOM	24.37	OOM	16.05	10.41	8.01	OOM	**1.10**	1.54	1.57
XL-n5649-k401	OOM	OOM	30.48	OOM	10.30	43.84	5.78	OOM	9.34	6.30	**5.16**
XL-n5774-k290	OOM	OOM	13.14	OOM	13.03	14.51	7.53	OOM	**4.62**	7.91	5.82
XL-n5902-k122	OOM	OOM	14.20	OOM	13.11	12.83	9.11	OOM	**6.14**	13.74	10.70
XL-n6034-k61	OOM	OOM	35.39	OOM	24.60	31.27	26.64	OOM	**8.68**	11.20	12.19
XL-n6168-k1922	OOM	OOM	116.85	OOM	**7.20**	40.60	11.24	OOM	11.53	12.27	11.74
XL-n6305-k1042	OOM	OOM	62.20	OOM	7.20	13.41	7.32	OOM	6.82	**5.33**	13.00
XL-n6445-k628	OOM	OOM	13.63	OOM	6.55	5.03	4.51	OOM	2.73	3.02	**2.28**
XL-n6588-k473	OOM	OOM	15.29	OOM	9.95	17.91	6.51	OOM	3.46	3.54	**2.97**
XL-n6734-k330	OOM	OOM	13.93	OOM	8.59	8.84	6.24	OOM	**3.00**	3.67	3.28
XL-n6884-k148	OOM	OOM	39.88	OOM	14.28	36.50	10.64	OOM	7.67	7.76	**6.70**
XL-n7037-k38	OOM	OOM	41.07	OOM	31.20	29.05	**9.00**	OOM	16.67	15.78	15.72
XL-n7193-k1683	OOM	OOM	68.72	OOM	8.87	8.67	5.43	OOM	7.36	**3.34**	17.78
XL-n7353-k1471	OOM	OOM	23.13	OOM	25.89	4.29	1.86	OOM	2.14	**0.84**	22.75
XL-n7516-k859	OOM	OOM	37.84	OOM	7.19	13.78	5.82	OOM	OOM	7.33	**3.84**
XL-n7683-k602	OOM	OOM	16.50	OOM	6.78	7.98	7.05	OOM	OOM	3.11	**2.95**
XL-n7854-k365	OOM	OOM	49.05	OOM	8.59	25.76	4.68	OOM	OOM	3.53	**2.90**
XL-n8028-k294	OOM	OOM	16.01	OOM	14.28	16.14	13.52	OOM	OOM	9.17	**9.09**
XL-n8207-k108	OOM	OOM	36.46	OOM	18.10	31.08	17.51	OOM	OOM	9.06	**8.17**
XL-n8389-k2028	OOM	OOM	77.49	OOM	**4.79**	17.87	6.20	OOM	OOM	7.60	6.15
XL-n8575-k1297	OOM	OOM	27.03	OOM	8.31	15.02	5.71	OOM	OOM	**3.02**	3.16
XL-n8766-k1032	OOM	OOM	11.26	OOM	6.33	8.50	6.72	OOM	OOM	4.08	**3.13**
XL-n8960-k634	OOM	OOM	11.14	OOM	8.21	12.23	16.70	OOM	OOM	2.97	**2.42**
XL-n9160-k379	OOM	OOM	20.50	OOM	16.79	52.11	13.60	OOM	OOM	8.02	**6.49**
XL-n9363-k209	OOM	OOM	22.91	OOM	19.09	20.25	13.00	OOM	OOM	6.34	**5.97**
XL-n9571-k55	OOM	OOM	38.07	OOM	30.78	29.34	50.74	OOM	OOM	18.32	**15.52**
XL-n9784-k2774	OOM	OOM	78.40	OOM	7.97	25.19	**2.14**	OOM	OOM	5.48	13.71
XL-n10001-k1570	OOM	OOM	23.22	OOM	5.66	7.22	4.93	OOM	OOM	**4.47**	7.31

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims in the abstract and introduction accurately reflect the paper's contributions and scope, with consistent and faithful descriptions throughout the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in Appendix G.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The current version of the paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide sufficient implementation and experimental details within the paper under reasonable space constraints, and we ensure that the reported results are reproducible.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code, data access instructions, and instructions for reproducing the experiments are available at `https://github.com/HamJewel/AR-GCN-VRP`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The training and test details, including data splits, hyperparameters, and optimizer settings, are fully specified in the main paper and the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We report average performance over test instances without error bars.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide sufficient information on the required computational resources in Section 5 to support reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in this paper conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All relevant external assets are properly credited and cited, and their licenses and terms of use are appropriately acknowledged and respected.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The released code is documented in the repository linked in the abstract.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.