

A Related work

A.1 Autoregressive Neural Solvers

Autoregressive neural solvers have emerged as a prominent research direction in the field of NCO. These methods typically employ an encoder-decoder architecture, where the encoder generates node embeddings, and the decoder dynamically produces a sequence of nodes.

Early works were based on Pointer Networks[30], gradually evolving from supervised learning to reinforcement learning training[1, 25]. Subsequently, the Transformer framework was introduced and has become a mainstream paradigm[16]. POMO[17] further leverages the symmetry of the VRP, optimizing the baseline calculation in RL training and enhancing the exploration diversity of the model. Subsequent research has largely focused on improving the Transformer framework [18, 35, 14, 15, 43, 27, 38]. However, these approaches predominantly adopt a heavy encoder light decoder structure, which poses challenges in handling large-scale problems. To enhance the capability of addressing large instances, some methods continue to build upon HELD. Examples include ELG[9], which integrates local search strategies; INVIT[7], which utilizes K-nearest neighbor multi-view embeddings to enhance local information; and ICAM[40], which introduces instance-conditional adaptation to better perceive problem scales. On the other hand, some studies[23, 5] have explored heavy decoder architectures. By dynamically capturing the relationships between remaining nodes during the decoding process, these methods demonstrate superior generalization performance on large-scale problems. However, their high computational demands, often necessitating training on labeled data, represent a significant limitation. Furthermore, decomposition-based methods[12, 36, 39, 24] warrant attention. These approaches decompose large-scale problems into multiple smaller subproblems for solving, and autoregressive solvers can be employed at the lower level to address these subproblems.

A.2 Multi-Task Learning for Solving VRPs

To enhance the generalizability of neural solvers across diverse VRP variants, several existing methods have proposed unified multi-task models. For instance, Wang et al.[31] employs a multi-armed bandit approach to achieve efficient training across multiple combinatorial optimization problems. Lin et al.[20] pre-train a Transformer backbone on the Traveling Salesman Problem and fine-tune it for specific VRP variants, thereby extending its applicability to a subset of VRP problems. These methods represent initial attempts at cross-problem learning but are limited in scope, focusing primarily on a small number of problems. Building on these foundations, MT-POMO [21] views VRP variants as combinations of distinct attributes, enabling unified representation and zero-shot generalization across multiple VRP problems, achieving notable performance on ten tasks. MVMoE [41] further boosts model capacity and multi-task performance by introducing a mixture-of-experts model. RouteFinder[2] enhances multi-task training efficiency and performance through mixed batch training and enables rapid adaptation to unseen tasks via efficient adapter layers. CaDA [19] improves the model’s ability to handle diverse tasks within a multi-task framework by incorporating constraint awareness and a dual-branch structure. However, these studies have predominantly focused on small-scale problems, lacking extensive exploration of large-scale generalization capabilities.

B Problem Definition and Settings

Our VRP variant problem settings follow the work of Zhou et al. [41]. Detailed settings are as follows:

- **Coordinates:** All node coordinates are uniformly sampled from the 2D space $[0, 1) \times [0, 1)$.
- **Demands:** For all customer nodes, demands are randomly sampled from the set $\{1, 2, \dots, 9\}$.
- **Vehicle Capacity:** Each vehicle has a fixed transportation capacity, which we set to 50 across all problem scales. The sum of demands for nodes visited by a vehicle must be less than or equal to its current capacity.
- **Open Routes:** In open vehicle routing problems, vehicles are not required to return to the depot. We use a binary flag vector to distinguish whether a path is open; setting it to 1 indicates an open path.

Table 4: 16 VRP variants with five constraints.

Constraint	Capacity	Open Route	Backhaul	Duration Limit	Time Window
CVRP	✓				
OVRP	✓	✓			
VRPB	✓		✓		
VRPL	✓			✓	
VRPTW	✓				✓
OVRPTW	✓	✓			✓
OVRPB	✓	✓	✓		
OVRPL	✓	✓		✓	
VRPBL	✓		✓	✓	
VRPBTW	✓		✓		✓
VRPLTW	✓			✓	✓
OVRPBL	✓	✓	✓	✓	
OVRPBTW	✓	✓	✓		✓
OVRPLTW	✓	✓		✓	✓
VRPBLTW	✓		✓	✓	✓
OVRPBLTW	✓	✓	✓	✓	✓

- **Backhauls:** In problems involving backhauls, a portion of the nodes have negative demands, referred to as backhaul nodes. We designate 20% of the customer nodes as backhaul nodes, whose demands are randomly sampled from $\{-1, -2, \dots, -9\}$.
- **Duration Limit:** Each vehicle has a maximum travel distance limit, which we set to 3.
- **Time Windows:** Customer node time windows are uniformly sampled, and service time is uniformly set to 0.2. Vehicles must visit nodes within their time windows; if a vehicle arrives early, it must wait until the earliest start time. The depot’s time window is set to $[0, 3]$, with a service time of 0. Vehicle speed is uniformly set to 1, meaning the time spent traveling on a path equals its distance divided by speed. Notably, for non-open problems, it is crucial to ensure that after visiting the last customer node, the vehicle can return to the depot within its latest allowable time window; this consideration is not necessary for open problems.

By combining different constraints, 16 VRP variants can be formed, as detailed in Table 4.

C Performance on Real-World Instances

Our test results on Real-World Instances datasets are presented in Tables 5, 6, and 7. All models are trained on instances of size $N = 100$. During inference, data augmentation is applied, and a greedy rollout strategy is adopted by default. Some results for comparative models are sourced from MVMoE [41] and RouteFinder [2].

D Licenses for Code and Datasets

The licenses for the codes and the datasets used in this work are listed in Table 8.

E Broader Impacts

This research contributes to the field of neural combinatorial optimization by employing advanced machine learning techniques to address large-scale and multi-variant VRP problems. We believe that the proposed multi-task knowledge distillation training framework, heavy decoder model, and Random Reordering Reconstruction (R3C) strategy can provide valuable insights and inspire subsequent work to explore more efficient and effective neural methods for solving large-scale and diverse VRP problems. As a general learning-based approach to solving the VRP, the proposed multi-task knowledge distillation training framework and R3C strategy do not inherently possess any specific potential negative social impacts.

Table 5: Results on small-scale CVRPLIB instances (Set-X) [28]. All models are trained on instances of size $N = 100$, and inference utilized data augmentation[17] and greedy rollout by default.

Set-X		POMO		POMO-MTL		MVMOE/4E		MVMOE/4E-L		MTL-KD	
Instance	Opt.	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap
X-n101-k25	27591	30138	9.231%	32482	17.727%	29361	6.415%	29015	5.161%	29058	5.317%
X-n106-k14	26362	39322	49.162%	27369	3.820%	27278	3.475%	27242	3.338%	26918	2.109%
X-n110-k13	14971	15223	1.683%	15151	1.202%	15089	0.788%	15196	1.503%	15407	2.912%
X-n115-k10	12747	16113	26.406%	14785	15.988%	13847	8.629%	13325	4.534%	13513	6.009%
X-n120-k6	13332	14085	5.648%	13931	4.493%	14089	5.678%	13833	3.758%	13657	2.438%
X-n125-k30	55539	58513	5.355%	60687	9.269%	58944	6.131%	58603	5.517%	57615	3.738%
X-n129-k18	28940	29246	1.057%	30332	4.810%	29802	2.979%	29457	1.786%	29309	1.275%
X-n134-k13	10916	11302	3.536%	11581	6.092%	11353	4.003%	11398	4.416%	11363	4.095%
X-n139-k10	13590	14035	3.274%	13911	2.362%	13825	1.729%	13800	1.545%	13911	2.362%
X-n143-k7	15700	16131	2.745%	16660	6.115%	16125	2.707%	16147	2.847%	15955	1.624%
X-n148-k46	43448	49328	13.533%	50782	16.880%	46758	7.618%	45599	4.951%	45463	4.638%
X-n153-k22	21220	32476	53.040%	26237	23.643%	23793	12.125%	23316	9.877%	23340	9.991%
X-n157-k13	16876	17660	4.646%	17510	3.757%	17650	4.586%	17410	3.164%	17161	1.689%
X-n162-k11	14138	14889	5.312%	14720	4.117%	14654	3.650%	14662	3.706%	14487	2.469%
X-n167-k10	20557	21822	6.154%	21399	4.096%	21340	3.809%	21275	3.493%	21053	2.413%
X-n172-k51	45607	49556	8.659%	56385	23.632%	51292	12.465%	49073	7.600%	47850	4.918%
X-n176-k26	47812	54197	13.354%	57637	20.549%	55520	16.121%	52727	10.280%	52476	9.755%
X-n181-k23	25569	37311	45.923%	26219	2.542%	26258	2.695%	26241	2.628%	25919	1.369%
X-n186-k15	24145	25222	4.461%	25000	3.541%	25182	4.295%	24836	2.862%	24711	2.344%
X-n190-k8	16980	18315	7.862%	18113	6.673%	18327	7.933%	18113	6.673%	17539	3.292%
X-n195-k51	44225	49158	11.154%	54090	22.306%	49984	13.022%	48185	8.954%	46301	4.694%
X-n200-k36	58578	64618	10.311%	61654	5.251%	61530	5.039%	61483	4.959%	60978	4.097%
X-n209-k16	30656	32212	5.076%	32011	4.420%	32033	4.492%	32055	4.564%	31536	2.871%
X-n219-k73	117595	133545	13.564%	119887	1.949%	121046	2.935%	120421	2.403%	118499	0.769%
X-n228-k23	25742	48689	89.142%	33091	28.549%	31054	20.636%	28561	10.951%	28156	9.378%
X-n237-k14	27042	29893	10.543%	28472	5.288%	28550	5.577%	28486	5.340%	27789	2.762%
X-n247-k50	37274	56167	50.687%	45065	20.902%	43673	17.167%	41800	12.143%	41106	10.281%
X-n251-k28	38684	40263	4.082%	40614	4.989%	41022	6.044%	40822	5.527%	39877	3.084%
Avg. Gap		16.629%		9.820%		6.884%		5.160%		4.025%	

Table 6: Results on VRPTW instances (Set-Solomon) [26]. All models are trained on instances of size $N = 100$, and inference utilized data augmentation[17] and greedy rollout by default.

Set-Solomon		POMO		POMO-MTL		MVMOE/4E		MVMOE/4E-L		MTL-KD	
Instance	Opt.	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap
R101	1637.7	1805.6	10.252%	1821.2	11.205%	1798.1	9.794%	1730.1	5.641%	1768.0	7.956%
R102	1466.6	1556.7	6.143%	1596.0	8.823%	1572.0	7.187%	1574.3	7.345%	1564.4	6.668%
R103	1208.7	1341.4	10.979%	1327.3	9.812%	1328.2	9.887%	1359.4	12.470%	1379.0	14.090%
R104	971.5	1118.6	15.142%	1120.7	15.358%	1124.8	15.780%	1098.8	13.100%	1074.1	10.561%
R105	1355.3	1506.4	11.149%	1514.6	11.754%	1479.4	9.157%	1456.0	7.433%	1465.2	8.109%
R106	1234.6	1365.2	10.578%	1380.5	11.818%	1362.4	10.352%	1353.5	9.627%	1346.1	9.031%
R107	1064.6	1214.2	14.052%	1209.3	13.592%	1182.1	11.037%	1196.5	12.391%	1193.9	12.145%
R108	932.1	1058.9	13.604%	1061.8	13.915%	1023.2	9.774%	1039.1	11.481%	1035.4	11.082%
R109	1146.9	1249.0	8.902%	1265.7	10.358%	1255.6	9.478%	1224.3	6.750%	1260.0	9.861%
R110	1068.0	1180.4	10.524%	1171.4	9.682%	1185.7	11.021%	1160.2	8.635%	1199.5	12.313%
R111	1048.7	1177.2	12.253%	1211.5	15.524%	1176.1	12.148%	1197.8	14.220%	1178.7	12.396%
R112	948.6	1063.1	12.070%	1057.0	11.427%	1045.2	10.183%	1044.2	10.082%	1057.2	11.448%
RC101	1619.8	2643.0	63.168%	1833.3	13.181%	1774.4	9.544%	1749.2	7.988%	1774.0	9.520%
RC102	1457.4	1534.8	5.311%	1546.1	6.086%	1544.5	5.976%	1556.1	6.771%	1588.4	8.989%
RC103	1258.0	1407.5	11.884%	1396.2	10.986%	1402.5	11.486%	1415.3	12.502%	1451.8	15.405%
RC104	1132.3	1261.8	11.437%	1271.7	12.311%	1265.4	11.755%	1264.2	11.649%	1241.6	9.653%
RC105	1513.7	1612.9	6.553%	1644.9	8.668%	1635.5	8.047%	1619.4	6.980%	1688.3	11.535%
RC106	1372.7	1539.3	12.137%	1552.8	13.120%	1505.0	9.638%	1509.5	9.968%	1468.8	7.001%
RC107	1207.8	1347.7	11.583%	1384.8	14.655%	1351.6	11.906%	1324.1	9.625%	1367.6	13.231%
RC108	1114.2	1305.5	17.169%	1274.4	14.378%	1254.2	12.565%	1247.2	11.939%	1232.3	10.600%
RC201	1261.8	2045.6	62.118%	1761.1	39.570%	1577.3	25.004%	1517.8	20.285%	1490.8	18.149%
RC202	1092.3	1805.1	65.257%	1486.2	36.062%	1616.5	47.990%	1480.3	35.520%	1336.5	22.356%
RC203	923.7	1470.4	59.186%	1360.4	47.277%	1473.5	59.521%	1479.6	60.182%	1237.8	34.005%
RC204	783.5	1323.9	68.973%	1331.7	69.968%	1286.6	64.212%	1232.8	57.342%	1066.0	36.056%
RC205	1154.0	1568.4	35.910%	1539.2	33.380%	1537.7	33.250%	1440.8	24.850%	1448.3	25.503%
RC206	1051.1	1707.5	62.449%	1472.6	40.101%	1468.9	39.749%	1394.5	32.671%	1354.7	28.884%
RC207	962.9	1567.2	62.758%	1375.7	42.870%	1442.0	49.756%	1346.4	39.831%	1235.9	28.352%
RC208	776.1	1505.4	93.970%	1185.6	52.764%	1107.4	42.688%	1167.5	50.437%	1064.0	37.096%
Avg. Gap		28.054%		21.380%		20.317%		18.490%		15.786%	

Table 7: Results on large-scale CVRPLIB instances (Set-X) [28]. All models are trained on instances of size $N = 100$, and inference utilized data augmentation[17] and greedy rollout by default.

Set-X		POMO		LEHD		POMO-MTL		MVMoe/4E		MVMOE/4E-L		RF-MVMOE		RF-TE		MTL-KD	
Instance	Opt.	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap	Obj.	Gap
X-n502-k39	69226	75617	9.232%	71438	3.195%	77284	11.640%	73533	6.222%	74429	7.516%	76338	10.274%	71791	3.705%	71124	2.742%
X-n513-k21	24201	30518	26.102%	25624	5.880%	28510	17.805%	32102	32.647%	31231	29.048%	32639	34.866%	28465	17.619%	25947	7.215%
X-n524-k153	154593	201877	30.586%	280556	81.480%	192249	24.358%	186540	20.665%	182392	17.982%	170999	10.612%	174381	12.800%	171306	10.811%
X-n536-k96	94846	106073	11.837%	103785	9.425%	106514	12.302%	109581	15.536%	108543	14.441%	105847	11.599%	103272	8.884%	101893	7.430%
X-n548-k50	86700	103093	18.908%	90644	4.549%	94562	9.068%	95894	10.604%	95917	10.631%	104289	20.287%	100956	16.443%	89169	2.848%
X-n561-k42	42717	49370	15.575%	44728	4.708%	47846	12.007%	56008	31.114%	51810	21.287%	53383	24.969%	49454	15.771%	45467	6.438%
X-n573-k30	50673	83545	64.871%	53482	5.543%	60913	20.208%	59473	17.366%	57042	12.569%	61524	21.414%	55952	10.418%	53466	5.512%
X-n586-k159	190316	229887	20.792%	232867	22.358%	208893	9.761%	215668	13.321%	214577	12.748%	212151	11.473%	205575	8.018%	200863	5.542%
X-n599-k92	108451	150572	38.839%	115377	6.386%	120333	10.956%	128949	18.901%	125279	15.517%	126578	16.714%	116560	7.477%	113513	4.668%
X-n613-k62	59535	68451	14.976%	62484	4.953%	67984	14.192%	82586	38.718%	74945	25.884%	73456	23.383%	67267	12.987%	63035	5.879%
X-n627-k43	62164	84434	35.825%	67568	8.693%	73060	17.528%	70987	14.193%	70905	14.061%	70414	13.271%	67572	8.700%	65755	5.777%
X-n641-k35	63682	75573	18.672%	68249	7.172%	72643	14.071%	75329	18.289%	72655	14.090%	71975	13.023%	70831	11.226%	67593	6.141%
X-n655-k131	106780	127211	19.134%	117532	10.069%	116988	9.560%	117678	10.206%	118475	10.952%	119057	11.497%	112202	5.078%	109748	2.780%
X-n670-k130	146332	208079	42.197%	220927	50.977%	190118	29.922%	197695	35.100%	183447	25.364%	168226	14.962%	168999	15.490%	161076	10.076%
X-n685-k75	68205	79482	16.534%	72946	6.951%	80892	18.601%	97388	42.787%	89441	31.136%	82269	20.620%	77847	14.137%	73249	7.395%
X-n701-k44	81923	97843	19.433%	86327	5.376%	92075	12.392%	98469	20.197%	94924	15.870%	90189	10.090%	89932	9.776%	85967	4.936%
X-n716-k35	43373	51381	18.463%	46502	7.214%	52709	21.525%	56773	30.895%	52305	20.593%	52250	20.467%	49669	14.516%	47012	8.390%
X-n733-k159	136187	159098	16.823%	149115	9.493%	161961	18.925%	178322	30.939%	167477	22.976%	156387	14.833%	148463	9.014%	142712	4.791%
X-n749-k98	77269	87786	13.611%	83439	7.985%	90582	17.229%	100438	29.985%	94497	22.296%	92147	19.255%	85171	10.227%	82295	6.505%
X-n766-k71	114417	135464	18.395%	131487	14.919%	144041	25.891%	152352	33.155%	136255	19.086%	130505	14.061%	129935	13.563%	123310	7.772%
X-n783-k48	72386	90289	24.733%	76766	6.051%	83169	14.897%	100383	38.677%	92960	28.423%	96336	33.087%	83185	14.919%	77332	6.833%
X-n801-k40	73305	124278	69.536%	77546	5.785%	85077	16.059%	91560	24.903%	87662	19.585%	87118	18.843%	86164	17.542%	78041	6.460%
X-n819-k171	158121	193451	22.344%	178558	12.925%	177157	12.039%	183599	16.113%	185832	17.525%	179596	13.581%	174441	10.321%	170672	7.938%
X-n837-k142	193737	237884	22.787%	207709	7.212%	214207	10.566%	229526	18.473%	221286	14.220%	230362	18.904%	208528	7.635%	203165	4.866%
X-n856-k95	88965	152528	71.447%	92936	4.464%	101774	14.398%	99129	11.425%	106816	20.065%	105801	18.924%	98291	10.483%	94079	5.748%
X-n876-k59	99299	119764	20.609%	104183	4.918%	116617	17.440%	119619	20.463%	114333	15.140%	114016	14.821%	107416	8.174%	105873	6.620%
X-n895-k37	53860	70245	30.421%	58028	7.739%	65587	21.773%	79018	46.710%	64310	19.402%	69099	28.294%	64871	20.444%	58606	8.812%
X-n916-k207	329179	399372	21.324%	385208	17.021%	361719	9.885%	383681	16.557%	374016	13.621%	373600	13.494%	352998	7.236%	346940	5.396%
X-n936-k151	132715	237625	79.049%	196547	48.097%	186262	40.347%	220926	66.466%	190407	43.471%	161343	21.571%	163162	22.942%	152094	14.602%
X-n957-k87	85465	130850	53.104%	90295	5.651%	98198	14.898%	113882	33.250%	105629	23.593%	123633	44.659%	102689	20.153%	90407	5.782%
X-n979-k58	118976	147687	24.132%	127972	7.561%	138092	16.067%	146347	23.005%	139682	17.404%	131754	10.740%	129952	9.225%	127650	7.291%
X-n1001-k43	72355	100399	38.759%	76689	5.990%	87660	21.153%	114448	58.176%	94734	30.929%	88969	22.962%	85929	18.760%	78833	8.953%
Avg. Gap		29.658%		12.836%		16.796%		26.408%		19.607%		18.795%		12.303%		6.655%	

Table 8: Licenses of Used Codes and Datasets

Type	Resource	License Type	URL / Reference
Code	MVMOE[41]	MIT License	https://github.com/RoyalSkye/Routing-MVMOE
Code	MT-POMO[21]	MIT License	https://github.com/FeiLiu36/MTNCO
Code	HGS-PyVRP[33]	MIT License	https://github.com/PyVRP/PyVRP
Code	OR-Tools[8]	Apache License	https://github.com/google/or-tools
Dataset	CVRPLIB(Set-X)[28]	Available for academic research use	http://vrp.galgos.inf.puc-rio.br/index.php/en/
Dataset	Solomon[26]	Available for academic research use	https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/