HYPERVOLUME-GUIDED ITERATIVE REINFORCEMENT LEARNING FOR MULTI-OBJECTIVE NEURAL COMBINATORIAL OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-objective combinatorial optimization requires balancing competing objectives in routing, packing, and other discrete decision problems. Preference-conditioned neural methods commonly use scalarized rewards to train a single policy to construct solutions across different objective trade-offs. However, generating high-quality solution sets remains challenging because scalarized rewards overlook candidates' unique coverage within a group, while using each collected batch for only one policy update limits further quality improvements under a fixed sampling budget. To address this challenge, we propose *Hypervolume-Guided Iterative Reinforcement Learning (HGI-RL)*, a training framework that combines marginal-HV advantage redistribution with repeated trajectory learning to improve solution-set quality. Specifically, exact marginal hypervolume contributions within each instance–preference candidate group redistribute positive advantages, incorporating candidates' unique coverage into the training signal. Moreover, the same trajectories and fixed redistributed advantages support multiple clipped proximal policy optimization (PPO) updates before new candidates are generated, allowing repeated learning from the collected solutions. To evaluate HGI-RL, experiments are conducted on multi-objective traveling salesman, capacitated vehicle routing, and knapsack problems. The results show that HGI-RL achieves competitive solution-set hypervolume.

1 INTRODUCTION

Multi-objective combinatorial optimization (MOCO) arises in routing, packing, and other discrete decision problems with conflicting objectives (Ehrgott, 2005; Zajac & Huber, 2021; Lust & Teghem, 2012). A route that minimizes distance may incur a higher cost, while a packing solution that favors one profit criterion may sacrifice another. Different preferences therefore favor different solutions, and the goal is to obtain a set that represents these trade-offs with both high quality and broad coverage.

Preference-conditioned neural methods learn a shared construction policy across problem instances and objective weights (Lin et al., 2022; Wang et al., 2024; Fan et al., 2025a; Chen et al., 2025). They commonly train on weighted sums of objectives and generate a solution set by evaluating multiple preferences. Policy optimization with multiple optima (POMO) provides several candidate solutions for each instance and preference, with their relative scalarized rewards guiding policy updates (Kwon et al., 2020). Other methods incorporate generated solutions and front information to promote complementary coverage (Chen et al., 2023b; Lu et al., 2025). We study how training can make better use of sampled candidates to improve the final solution set under a fixed sampling budget, focusing on two challenges.

The first challenge is to distinguish individual quality from contribution to a set. Candidates with similar scalarized rewards may duplicate coverage already supplied by the group or extend it into a new region. Scalarized rewards alone do not distinguish these roles. Hypervolume (HV) measures the objective-space volume covered relative to a reference point; its decrease when a candidate is removed quantifies that candidate's exclusive contribution (Audet et al., 2021; Chen et al., 2022). This

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

contribution captures the unique coverage that scalarized rewards overlook and provides additional feedback for improving solution-set quality.

The second challenge is to learn sufficiently from the collected solutions. REINFORCE-style POMO training typically applies one policy update to a sampled batch before constructing new trajectories (Williams, 1992; Kwon et al., 2020). Using each batch only once limits further quality improvements under a fixed sampling budget. Repeated updates on the same trajectories allow the policy to learn further from observed solution quality and coverage before generating new candidates.

To address these challenges, we propose Hypervolume-Guided Iterative Reinforcement Learning (HGI-RL), a training framework that combines marginal-HV advantage redistribution with repeated trajectory learning. Within each instance–preference group, exact marginal HV contributions measure each candidate's exclusive coverage and guide positive-advantage redistribution while preserving the original scalarized reward, non-positive advantages, and total positive advantage mass. The training signal thus combines scalarized solution quality with each candidate's contribution to group coverage.

HGI-RL uses proximal policy optimization (PPO) for its policy updates (Schulman et al., 2017). The redistributed advantages remain fixed during multiple clipped updates on the same trajectories. The updated policy then generates a new batch, and the candidates' contributions are recomputed for the next iteration.

Our contributions are threefold:

- We propose HGI-RL to integrate group-level coverage feedback into repeated policy learning from sampled trajectories.
- We introduce an advantage redistribution rule based on exact exclusive HV contributions within each instance–preference group, preserving non-positive advantages and the total positive advantage mass.
- We evaluate HGI-RL on multi-objective traveling salesman, capacitated vehicle routing, and knapsack problems using a preference-conditioned attention network (PCAN), achieving competitive solution-set HV. We also apply the framework to WE-CA and CNH and compare REINFORCE, PPO, and HGI-RL in a fixed-network ablation.

2 Related Work

Classical MOCO methods approximate Pareto fronts through decomposition, evolutionary search, or Pareto local search (Deb et al., 2002; Zhang & Li, 2007; Shi et al., 2020). Neural methods either construct solutions directly or guide iterative search. Search-augmented approaches adapt model parameters to a test instance (Wu et al., 2024), learn neighborhood-search decisions (Zheng et al., 2024; Zhang et al., 2026), or initialize evolutionary optimization with neural solutions (Wu et al., 2025).

Representation and generation. Preferences can enter a construction policy through decoder parameters, as in PMOCO and WSDAM (Lin et al., 2022; Ye et al., 2022), or through learned representations, as in CNH and Neat Weight Embedding (Fan et al., 2025a; Chen et al., 2025). Extensions address solution structures and decoding (Li et al., 2023; Su et al., 2024; Li et al., 2026; Rydin et al., 2026), while retention-based models reduce generation costs (Wei et al., 2025). These architectural choices determine how a policy represents each scalarized subproblem.

Training and adaptation. Parameter transfer reuses policies across neighboring objective weights (Li et al., 2021; Gao et al., 2022), while meta-learning supports adaptation across preferences and tasks (Zhang et al., 2023; Chen et al., 2023a; Fu et al., 2026). Policy-population evolution (Zhang et al., 2021) and adversarial training (Liu et al., 2026) provide further training strategies. POCCO combines conditional computation with optimization based on pairwise preferences between solutions (Fan et al., 2025b). At the update level, REINFORCE learns from sampled rewards (Williams, 1992; Bello et al., 2016; Kool et al., 2019), POMO supplies multiple starts and group-relative baselines (Kwon et al., 2020), and PPO enables clipped trajectory reuse (Schulman

et al., 2017). HGI-RL retains scalarized rewards and uses exclusive HV contributions to adjust positive advantages within this PPO procedure.

Pareto-set coordination. NHDE jointly encodes the instance and current Pareto front through heterogeneous graph attention, using preferences and diversity information to guide subsequent solutions (Chen et al., 2023b). CDE combines node- and solution-level context with HV-based rewards and an HV residual update, capturing local and non-local set information during sequential generation (Lu et al., 2025). Both connect candidates generated across preferences. Preference adjustment instead learns which objective weights to query from current solutions and HV gains (Wang et al., 2023).

The feedback in HGI-RL is local to candidates sampled for one instance and preference; NHDE and CDE coordinate construction using solutions generated across preferences.

3 Problem Formulation

Let $I \sim \mathcal{D}$ be an instance of size n sampled from a training distribution $\mathcal{D}$, and let $x \in \mathcal{X}(I)$ be a feasible solution. Its $m \geq 2$ objective values are $\mathbf{f}(x; I) = (f_1(x; I), \ldots, f_m(x; I))$. A preference $\mathbf{w}$ lies on the probability simplex Δ^{m-1} and satisfies $w_\ell \geq 0$ and $\sum_\ell w_\ell = 1$. For minimization, the scalarized reward is

$$R(x; I, \mathbf{w}) = -\sum_{\ell=1}^{m} w_\ell f_\ell(x; I), \tag{1}$$

and for maximization the sign is reversed. Task definitions and construction states are provided in Appendix A. An autoregressive policy π_θ, parameterized by θ, constructs a trajectory $\tau = (a_1, \ldots, a_T)$, where a_t is the action at step t and T is the number of steps before termination. The state s_t contains the instance and partial solution. Conditional on the designated POMO start, the trajectory log-probability is

$$\log \pi_\theta(\tau \mid I, \mathbf{w}) = \sum_{t=1}^{T} \log \pi_\theta(a_t \mid s_t, \mathbf{w}). \tag{2}$$

Fixed initialization actions have probability one, and post-termination padding is excluded from this sum. For a batch $\mathcal{B}$ of B instances, let $b \in \{1, \ldots, B\}$ index instances and $k \in \{1, \ldots, K\}$ index POMO candidates. One preference $\mathbf{w}_b$ is sampled for each instance I_b. Candidate k produces solution $x_{b,k}$ with objective vector $\mathbf{f}_{b,k} = \mathbf{f}(x_{b,k}; I_b)$ and reward $R_{b,k} = R(x_{b,k}; I_b, \mathbf{w}_b)$. Its group-relative advantage is

$$A^{\text{raw}}_{b,k} = R_{b,k} - \frac{1}{K} \sum_{j=1}^{K} R_{b,j}. \tag{3}$$

For PPO and HGI-RL, the collected advantages are normalized jointly over the batch and candidate dimensions,

$$A_{b,k} = \frac{A^{\text{raw}}_{b,k} - \mu_{\mathcal{B}}}{s_{\mathcal{B}} + 10^{-8}}, \tag{4}$$

where $\mu_{\mathcal{B}}$ and $s_{\mathcal{B}}$ are the batch mean and sample standard deviation. HGI-RL reweights the positive entries of A using marginal-HV contributions.

4 HGI-RL Framework

Figure 1 summarizes the training cycle: candidate generation, objective normalization, HV-guided advantage redistribution, and repeated policy updates. The following subsections specify each stage and the inference procedure.

4.1 Candidate Generation

For each training instance I_b, we sample one preference $\mathbf{w}_b$ and use a frozen rollout policy $\pi_{\theta_{\text{old}}}$ to construct K POMO trajectories. All candidates in the group share the same instance and preference.

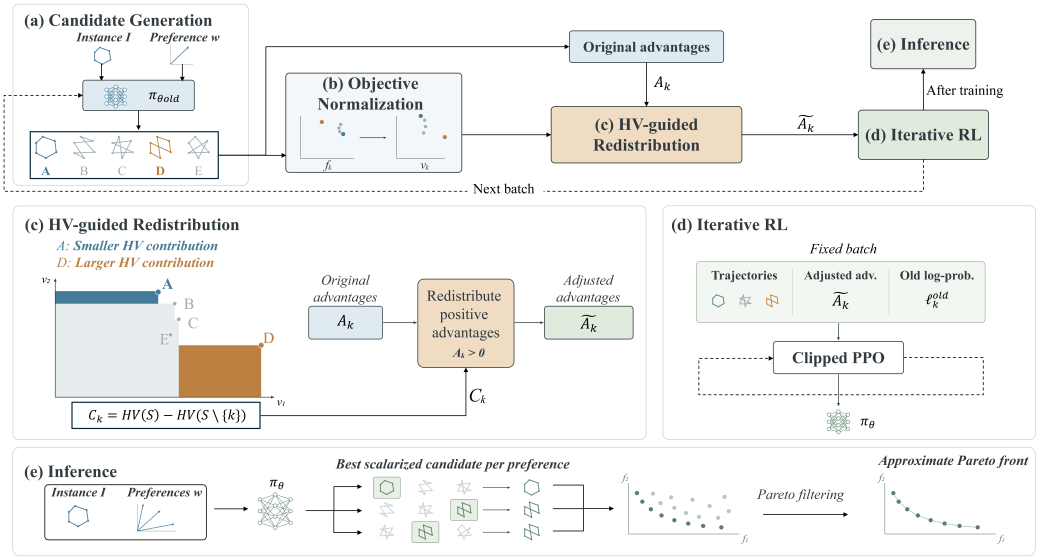

Figure 1: The HGI-RL framework. (a–b) The rollout policy generates candidates for one instance–preference pair, and their objectives are normalized. (c) Blue and orange regions illustrate the exclusive HV contributions of candidates A and D. These contributions guide positive-advantage redistribution, preserving non-positive advantages and the sum of positive advantages. (d) Five clipped PPO updates reuse fixed trajectories, redistributed advantages, and old log-probabilities before the updated policy generates the next batch. (e) At inference, the best scalarized candidate for each preference is retained, followed by Pareto filtering to obtain an approximate Pareto front. The instance index b is omitted for clarity.

We store their sampled actions, objective vectors, and complete-trajectory log-probabilities together with the instances and preferences.

4.2 Objective Normalization

To compare candidates within each instance–preference group, we first normalize their objective values. The K objective vectors for one problem are represented in a common minimization cost space. Let

$$\mathbf{c}_{b,k} = \begin{cases} \mathbf{f}_{b,k}, & \text{for minimization,} \\ -\mathbf{f}_{b,k}, & \text{for maximization.} \end{cases} \tag{5}$$

Define the component-wise minimum and maximum over the full candidate group,

$$\mathbf{z}_b = \min_k \mathbf{c}_{b,k}, \qquad \mathbf{n}_b = \max_k \mathbf{c}_{b,k}, \tag{6}$$

where the operations are component-wise. With $\delta = 10^{-6}$,

$$\mathbf{s}_b = \max(\mathbf{n}_b - \mathbf{z}_b, \delta), \qquad \mathbf{r}_b = \mathbf{n}_b + \mathbf{s}_b. \tag{7}$$

Using component-wise division and maximum, the normalized improvement vector is

$$\mathbf{v}_{b,k} = \max\left(\frac{\mathbf{r}_b - \mathbf{c}_{b,k}}{\mathbf{r}_b - \mathbf{z}_b}, 0\right). \tag{8}$$

The geometry is computed once from the complete candidate set and frozen for every leave-one-out comparison. It is used only for advantage redistribution; the final evaluation HV continues to use the fixed benchmark bounds shared by all methods.

4.3 HV-GUIDED REDISTRIBUTION

The scalarized rewards yield the normalized group-relative advantages $A_{b,k}$ in Equations (3)–(4). These advantages compare candidates under the sampled preference. Marginal HV contributions measure the coverage lost when a candidate is removed. We use these contributions to redistribute positive advantages within each group.

4.3.1 MARGINAL HV

Each candidate defines an anchored box $[\mathbf{0}, \mathbf{v}_{b,k}]$. Let S_b denote the full indexed candidate collection for problem b, and let $\mathrm{HV}(S_b)$ be the volume of the union of these boxes. The exclusive HV contribution of trajectory k is

$$C_{b,k} = \mathrm{HV}(S_b) - \mathrm{HV}(S_b \setminus \{k\}). \tag{9}$$

Its normalized contribution is

$$p_{b,k} = \begin{cases} C_{b,k} / \sum_j C_{b,j}, & \sum_j C_{b,j} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

We retain all candidates in Equation (9), including dominated points and exact duplicates. A dominated point has zero exclusive contribution while its dominator is present, but can retain coverage after that dominator is removed and thus affect the dominator's contribution. Duplicate candidates receive zero exclusive contribution because another copy preserves their coverage.

For two objectives, the implementation partitions one objective axis at the candidate coordinates and assigns each interval's exclusive area to its contributing candidate. For three objectives, it partitions two axes and accumulates exclusive volume over the resulting cells. Directly accumulating exclusive area or volume avoids subtracting nearly equal total HV values. Accumulation uses double precision, and the three-objective computation processes cells in blocks to limit temporary memory. Contributions are exact up to floating-point precision.

4.3.2 ADVANTAGE REDISTRIBUTION

We reweight trajectories with positive scalarized advantages. Let $\mathcal{P}_b = \{k : A_{b,k} > 0\}$ and define their mean normalized contribution,

$$\bar{p}_b^{+} = \frac{1}{\max(|\mathcal{P}_b|, 1)} \sum_{k \in \mathcal{P}_b} p_{b,k}. \tag{11}$$

The mean is zero when $\mathcal{P}_b$ is empty. Let $\mathcal{G}$ contain the groups eligible for redistribution: they have at least two positive advantages, sufficient positive advantage sum, a positive finite mean contribution, and nonconstant contributions among those candidates. Appendix C.1 gives the numerical thresholds. For $k \in \mathcal{P}_b$, define the initial weight

$$q_{b,k} = 1 + \frac{p_{b,k}}{p_{b,k} + \bar{p}_b^{+} + 10^{-8}}. \tag{12}$$

The redistributed advantage is

$$\widetilde{A}_{b,k} = \begin{cases} A_{b,k} q_{b,k} \dfrac{\sum_{j \in \mathcal{P}_b} A_{b,j}}{\sum_{j \in \mathcal{P}_b} A_{b,j} q_{b,j}}, & b \in \mathcal{G},\ k \in \mathcal{P}_b, \\ A_{b,k}, & \text{otherwise.} \end{cases} \tag{13}$$

Groups outside $\mathcal{G}$ retain their original advantages.

Equation (13) preserves non-positive advantages, the sign of every advantage, and the sum of positive advantages:

$$\sum_{k \in \mathcal{P}_b} \widetilde{A}_{b,k} = \sum_{k \in \mathcal{P}_b} A_{b,k}. \tag{14}$$

Within an active group, a larger exclusive contribution gives a larger multiplicative adjustment. The adjustment depends on contribution, while the original advantage retains the comparison of scalarized quality.

The contribution is measured within one instance–preference group; final evaluation HV uses the outputs retained across preferences, as described in Section 4.5.

4.4 ITERATIVE RL

HGI-RL performs five PPO updates on each sampled batch, replaying the stored action sequences with fixed redistributed advantages. Write $\ell^{\theta}_{b,k} = \log \pi_\theta(\tau_{b,k} \mid I_b, \mathbf{w}_b)$ for the current trajectory log-probability and $\ell^{\mathrm{old}}_{b,k} = \log \pi_{\theta_{\mathrm{old}}}(\tau_{b,k} \mid I_b, \mathbf{w}_b)$ for the stored rollout value. The complete-trajectory likelihood ratio is

$$\begin{aligned} \Delta\ell_{b,k}(\theta) &= \ell^{\theta}_{b,k} - \ell^{\mathrm{old}}_{b,k}, \\ r_{b,k}(\theta) &= \exp[\Delta\ell_{b,k}(\theta)]\,. \end{aligned} \tag{15}$$

Given the fixed redistributed advantage $\widetilde{A}_{b,k}$ from Section 4.3, define

$$\begin{aligned} u_{b,k}(\theta) &= r_{b,k}(\theta)\widetilde{A}_{b,k}, \\ u^{\mathrm{clip}}_{b,k}(\theta) &= \mathrm{clip}(r_{b,k}(\theta), 1-\epsilon, 1+\epsilon)\,\widetilde{A}_{b,k}. \end{aligned} \tag{16}$$

The training loss is

$$\mathcal{L}_{\mathrm{PPO}}(\theta) = -\frac{1}{BK}\sum_{b=1}^{B}\sum_{k=1}^{K} \min\Big(u_{b,k}, u^{\mathrm{clip}}_{b,k}\Big) - \eta\mathcal{H}(\pi_\theta), \tag{17}$$

where $\mathrm{clip}(x, l, u) = \min(\max(x, l), u)$, $\epsilon = 0.2$, $\eta = 0.01$, and $\mathcal{H}$ is the action entropy defined in Appendix C.2. Stored rollout log-probabilities also remain fixed; current-policy probabilities and action entropy are recomputed at each update. PPO uses $\widetilde{A} = A$; HGI-RL uses Equation (13). After five updates, the updated policy collects a new batch, and its candidate contributions are computed for the next iteration.

Computational cost. HGI-RL and PPO use the same batch size, sampled-instance budget, and five optimizer steps per batch, with each method sampling from its own policy. HGI-RL computes and detaches the contributions once per batch, adding $O(BK^2)$ work for two objectives and $O(BK^3)$ for three objectives. This computation uses the collected objective vectors without additional rollouts or backward passes. It is confined to training; both methods use the same inference procedure. The REINFORCE control takes one optimizer step per batch (Appendix C.3).

4.5 INFERENCE

At evaluation, Q fixed preferences are solved independently. For each preference, the candidate with the largest scalarized reward is retained across the configured POMO starts and geometric transformations. The resulting objective vectors are Pareto filtered and evaluated by normalized HV using fixed benchmark bounds. Appendix B defines these quantities, and Appendix D details the backbone and evaluation protocol.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Problems and training protocol. We evaluate Bi-TSP, Bi-CVRP, and Tri-TSP at sizes 20/50/100, and Bi-KP at sizes 50/100/200. PCAN is the default policy network, with preference-conditioned attention and POMO multi-start sampling (Kool et al., 2019; Kwon et al., 2020). Models are trained for 200 epochs with 100,000 new instances per epoch and batch size 64, using the weighted-sum reward in Equation (1). Training samples one preference uniformly from the simplex per instance and uses $K = n$ POMO starts. Architecture and optimizer details are given in Appendix D.

Evaluation and reporting. Evaluation uses 200 fixed instances per size, 101 Das–Dennis preferences for two objectives, and 105 for three objectives (Das & Dennis, 1998). We report normalized HV and total evaluation time on the same RTX 4090-equipped machine. Times are single-run measurements. +Aug denotes 64 transformations for Bi-TSP, 8 for Bi-CVRP, and 512 for Tri-TSP; KP uses no augmentation. Internal comparisons use epoch-200 checkpoints and the same evaluation procedure (Section 4.5).

Table 1: Normalized HV, relative HV gap, and recorded evaluation time on bi-objective TSP and CVRP. Gap is $(\mathrm{HV}_{\mathrm{ref}} - \mathrm{HV})/\mathrm{HV}_{\mathrm{ref}} \times 100\%$, computed from the displayed HV values with PCAN+HGI-RL+Aug as the reference for each size; negative gaps indicate higher HV than the reference. Times are totals for 200 test instances measured on the same RTX 4090-equipped machine (S/M/H: seconds/minutes/hours); seconds are rounded to whole numbers, and values above 60 are converted to the next unit. Bold and underline mark the highest and second-highest distinct displayed HV values, including ties, and their corresponding gaps. Standard augmentation uses 64 transformations for TSP and 8 for CVRP. Dashes indicate unreported values.

Method	Bi-TSP20			Bi-TSP50			Bi-TSP100		
	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
WS-LKH	0.6270	0.02%	4.0M	**0.6415**	**0.00%**	43.2M	**0.7090**	**-0.18%**	3.9H
MOEA/D	0.6241	0.48%	41.1M	0.6316	1.54%	43.3M	0.6899	2.52%	1.4H
NSGA-II	0.6258	0.21%	2.6H	0.6120	4.60%	2.8H	0.6692	5.44%	4.8H
PPLS/D-C	0.6256	0.24%	10.5M	0.6282	2.07%	1.5H	0.6844	3.29%	7.3H
DRL-MOA	0.6257	0.22%	5S	0.6360	0.86%	6S	0.6970	1.51%	15S
MDRL	0.6271	0.00%	3S	0.6364	0.80%	5S	0.6969	1.53%	12S
EMNH	0.6271	0.00%	2S	0.6364	0.80%	6S	0.6969	1.53%	11S
PMOCO	0.6259	0.19%	3S	0.6351	1.00%	5S	0.6957	1.70%	13S
CNH	0.6270	0.02%	8S	0.6387	0.44%	9S	0.7019	0.82%	21S
WE-CA	0.6270	0.02%	2S	0.6391	0.37%	4S	0.7039	0.54%	13S
PCAN+HGI-RL	**0.6272**	**-0.02%**	2S	0.6400	0.23%	3S	0.7052	0.35%	12S
MDRL+Aug	0.6271	0.00%	20S	0.6408	0.11%	1.7M	0.7022	0.78%	11.9M
EMNH+Aug	0.6271	0.00%	20S	0.6408	0.11%	1.5M	0.7023	0.76%	11.7M
PMOCO+Aug	0.6270	0.02%	41S	0.6395	0.31%	2.5M	0.7016	0.86%	12.6M
CNH+Aug	0.6271	0.00%	54S	0.6410	0.08%	3.3M	0.7054	0.32%	13.2M
WE-CA+Aug	0.6271	0.00%	38S	0.6412	0.05%	2.5M	0.7067	0.14%	12.3M
PCAN+HGI-RL+Aug	0.6271	0.00%	28S	**0.6415**	**0.00%**	2.3M	0.7077	0.00%	11.8M

Method	Bi-CVRP20			Bi-CVRP50			Bi-CVRP100		
	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
MOEA/D	0.4255	1.09%	1.5H	0.4000	2.56%	1.7H	0.3953	3.21%	3.1H
NSGA-II	0.4275	0.63%	3.4H	0.3896	5.09%	3.9H	0.3620	11.36%	6.2H
PPLS/D-C	0.4287	0.35%	48.5M	0.4007	2.39%	4.5H	0.3946	3.38%	23.9H
DRL-MOA	0.4287	0.35%	6S	0.4076	0.71%	8S	0.4055	0.71%	22S
MDRL	0.4291	0.26%	7S	0.4082	0.56%	6S	0.4056	0.69%	20S
EMNH	0.4299	0.07%	6S	0.4098	0.17%	8S	0.4072	0.29%	21S
PMOCO	0.4267	0.81%	5S	0.4036	1.68%	6S	0.3913	4.19%	19S
CNH	0.4287	0.35%	9S	0.4087	0.44%	9S	0.4065	0.47%	21S
WE-CA	0.4293	0.21%	3S	0.4090	0.37%	5S	0.4072	0.29%	15S
PCAN+HGI-RL	0.4298	0.09%	2S	0.4100	0.12%	4S	0.4075	0.22%	14S
MDRL+Aug	0.4294	0.19%	6S	0.4092	0.32%	24S	0.4072	0.29%	2.4M
EMNH+Aug	**0.4302**	**0.00%**	7S	**0.4106**	**-0.02%**	25S	0.4079	0.12%	2.2M
PMOCO+Aug	0.4294	0.19%	10S	0.4080	0.61%	25S	0.3969	2.82%	2.5M
CNH+Aug	0.4299	0.07%	14S	0.4101	0.10%	30S	0.4077	0.17%	2.0M
WE-CA+Aug	0.4301	0.02%	7S	0.4104	0.02%	24S	0.4082	0.05%	1.9M
PCAN+HGI-RL+Aug	**0.4302**	**0.00%**	6S	0.4105	0.00%	21S	**0.4084**	**0.00%**	1.8M

Comparisons. We compare PCAN+HGI-RL with weighted-sum solvers (WS-LKH and WS-DP), evolutionary methods (MOEA/D and NSGA-II), Pareto local search (PPLS/D-C), and neural construction methods. The neural baselines include multi-model methods (DRL-MOA, MDRL, and EMNH) and single-policy methods (PMOCO, CNH, and WE-CA). The former use a collection of models to cover different preferences, whereas the latter represent these trade-offs with one policy. Appendix E.1 identifies their sources. PCAN is a compact policy network that incorporates preferences through encoder input features and uses a decoder without separate preference injection. It has approximately 1.38M parameters, compared with 1.47M for WE-CA and 1.63M for CNH. HGI-RL adds no trainable parameters to the policy network. Appendix D.1 details the architecture, and Appendix E reports model sizes, and Table 4 reports the application of HGI-RL to other networks.

Table 2: Normalized HV, relative HV gap, and recorded evaluation time on bi-objective KP and three-objective TSP. Gap and Time follow Table 1. The reference is PCAN+HGI-RL for KP and PCAN+HGI-RL+Aug for Tri-TSP. KP has no augmentation; Tri-TSP uses 512 transformations. Time units and bold/underline conventions follow Table 1.

Method	Bi-KP50			Bi-KP100			Bi-KP200		
	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
WS-DP	**0.3561**	**0.00%**	8.4M	0.4532	0.02%	1.1H	0.3601	0.00%	3.8H
MOEA/D	0.3540	0.59%	34.7M	0.4508	0.55%	53.4M	0.3581	0.56%	1.6H
NSGA-II	0.3547	0.39%	2.9H	0.4520	0.29%	4.4H	0.3590	0.31%	5.3H
PPLS/D-C	0.3528	0.93%	6.7M	0.4480	1.17%	24.7M	0.3541	1.67%	55.8M
DRL-MOA	0.3559	0.06%	5S	0.4531	0.04%	10S	0.3601	0.00%	39S
MDRL	0.3530	0.87%	4S	0.4532	0.02%	12S	0.3601	0.00%	47S
EMNH	**0.3561**	**0.00%**	3S	**0.4535**	**-0.04%**	12S	**0.3603**	**-0.06%**	46S
PMOCO	0.3552	0.25%	4S	0.4523	0.22%	12S	0.3595	0.17%	50S
CNH	0.3556	0.14%	8S	0.4527	0.13%	15S	0.3598	0.08%	45S
WE-CA	**0.3561**	**0.00%**	3S	0.4533	0.00%	10S	0.3601	0.00%	41S
PCAN+HGI-RL	**0.3561**	**0.00%**	3S	0.4533	0.00%	10S	0.3601	0.00%	40S
Method	Tri-TSP20			Tri-TSP50			Tri-TSP100		
	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
WS-LKH	**0.4712**	**0.00%**	5.9M	**0.4440**	**-0.11%**	49.6M	**0.5076**	**-0.57%**	4.8H
MOEA/D	0.4702	0.21%	55.0M	0.4314	2.73%	57.5M	0.4511	10.62%	1.9H
NSGA-II	0.4238	10.06%	3.9H	0.2858	35.56%	3.5H	0.2824	44.05%	6.5H
PPLS/D-C	0.4698	0.30%	40.5M	0.4174	5.89%	1.9H	0.4376	13.30%	9.9H
DRL-MOA	0.4699	0.28%	4S	0.4303	2.98%	5S	0.4806	4.78%	15S
MDRL	0.4699	0.28%	5S	0.4317	2.66%	6S	0.4852	3.86%	12S
EMNH	0.4699	0.28%	3S	0.4324	2.50%	6S	0.4866	3.59%	11S
PMOCO	0.4693	0.40%	5S	0.4315	2.71%	6S	0.4858	3.74%	14S
CNH	0.4698	0.30%	5S	0.4358	1.74%	9S	0.4931	2.30%	19S
WE-CA	0.4707	0.11%	2S	0.4389	1.04%	4S	0.4976	1.41%	13S
PCAN+HGI-RL	0.4708	0.08%	2S	0.4402	0.74%	4S	0.4995	1.03%	12S
MDRL+Aug	**0.4712**	**0.00%**	2.9M	0.4408	0.61%	18.3M	0.4958	1.76%	1.8H
EMNH+Aug	**0.4712**	**0.00%**	2.6M	0.4418	0.38%	18.1M	0.4973	1.47%	1.9H
PMOCO+Aug	**0.4712**	**0.00%**	5.1M	0.4409	0.59%	20.2M	0.4956	1.80%	2.1H
CNH+Aug	0.4704	0.17%	8.2M	0.4409	0.59%	23.7M	0.4996	1.01%	2.4H
WE-CA+Aug	**0.4712**	**0.00%**	5.1M	0.4431	0.09%	20.8M	0.5036	0.22%	1.7H
PCAN+HGI-RL+Aug	**0.4712**	**0.00%**	3.8M	0.4435	0.00%	18.9M	0.5047	0.00%	1.6H

5.2 Main Results

Solution quality. Tables 1 and 2 cover twelve problem–size settings. PCAN+HGI-RL matches or improves on WE-CA across the routing tasks at the reported precision. On Tri-TSP100, for example, its HV increases from WE-CA's 0.4976/0.5036 to 0.4995/0.5047 without/with augmentation. Augmented Bi-CVRP20 and Bi-CVRP100 also attain the highest displayed HV among the compared methods, with a tie on Bi-CVRP20. On KP, PCAN+HGI-RL matches WE-CA but remains below EMNH at sizes 100 and 200. Augmentation slightly lowers its Bi-TSP20 HV, so its benefit is not uniform.

Evaluation time. On Bi-TSP50, augmented PCAN+HGI-RL matches WS-LKH's HV of 0.6415 in 2.3M versus 43.2M, approximately $19\times$ faster; EMNH+Aug takes 1.5M with a lower HV of 0.6408. On Bi-CVRP50 and Bi-CVRP100, PCAN+HGI-RL has the shortest reported evaluation times among the neural methods both without augmentation (4S and 14S) and with augmentation (21S and 1.8M). On Tri-TSP50, its augmented variant raises HV from WE-CA+Aug's 0.4431 to 0.4435 while reducing time from 20.8M to 18.9M. Across the 20 matched configurations detailed in Appendix D.4, its full evaluation time is lower than WE-CA's.

5.3 Training-Rule Ablation

Table 3 compares REINFORCE, PPO, and HGI-RL using the same PCAN architecture, training data distribution, and sampling and inference budgets. HGI-RL retains the PPO update settings and adds marginal-HV redistribution of positive advantages. Detailed settings are given in Appendix C.3.

PPO improves over REINFORCE on all four reported tasks. HGI-RL further improves HV over PPO by 0.0003, 0.0002, and 0.0009 on Bi-TSP50, Bi-CVRP50, and Tri-TSP50, respectively. On Bi-KP100, PPO and HGI-RL tie at 0.4533 at the reported precision.

Table 3: Training-rule ablation on size-50 routing problems and size-100 KP. Values are mean normalized HV over 200 test instances without augmentation; higher is better. All variants use PCAN and epoch-200 checkpoints.

Training rule	Bi-TSP50	Bi-CVRP50	Bi-KP100	Tri-TSP50
REINFORCE	0.6394	0.4094	0.4530	0.4390
PPO	0.6397	0.4098	0.4533	0.4393
HGI-RL	0.6400	0.4100	0.4533	0.4402

Application to other policy networks. We also apply HGI-RL to WE-CA and CNH, retaining their architectures and training each from scratch for 200 epochs with the same evaluation preference grids and inference candidate budgets as its corresponding baseline. Both networks improve HV on Bi-TSP50 and Tri-TSP50, with and without augmentation. On Tri-TSP50, HGI-RL increases CNH's HV from 0.4358 to 0.4399 without augmentation and from 0.4409 to 0.4435 with augmentation. WE-CA also improves across all four settings, as shown in Table 4. These results show that the benefits of HGI-RL extend beyond the default PCAN architecture.

Table 4: Application of HGI-RL to WE-CA and CNH on 50-node TSP. Normalized HV; higher is better. Bi-TSP uses 64 augmentations and Tri-TSP uses 512.

Method	Bi-TSP50	Bi-TSP50+Aug	Tri-TSP50	Tri-TSP50+Aug
WE-CA	0.6391	0.6412	0.4389	0.4431
WE-CA+HGI-RL	0.6399	0.6415	0.4396	0.4434
CNH	0.6387	0.6410	0.4358	0.4409
CNH+HGI-RL	0.6396	0.6415	0.4399	0.4435

6 Limitations and Conclusion

HGI-RL uses contributions from one instance–preference group. Overlap within the group can leave few candidates with nonzero contributions, reducing opportunities for redistribution. The contribution calculation also becomes more costly as the number of objectives grows. Our experiments cover two and three objectives; coordinating contributions across preferences and extending the computation to more objectives remain directions for future work.

HGI-RL incorporates each candidate's exclusive coverage into positive advantages and reuses this feedback during PPO updates. The fixed-network ablation shows HV gains over PPO on the three routing tasks, and applying HGI-RL to WE-CA and CNH improves their results on 50-node TSP. With PCAN, the framework yields competitive solution sets and short evaluation times across the tested routing and packing problems.

AI USE STATEMENT

Writing assistance. Generative AI tools assisted with refining the manuscript wording and LaTeX formatting, including parts of the appendix. The authors reviewed and revised the AI-assisted text and take responsibility for its accuracy.

Literature search. Generative AI tools assisted with identifying and organizing related work on neural multi-objective combinatorial optimization and reinforcement learning. The authors are responsible for verifying the selected references against their original sources and for the final content of the paper.

ETHICS STATEMENT

This work studies multi-objective combinatorial optimization using synthetic benchmark instances. The experiments do not involve human participants or personal data.

REPRODUCIBILITY STATEMENT

Appendix A specifies the problem generators and construction states. Appendices B and C define evaluation HV, advantage redistribution, and policy updates, including numerical safeguards. Appendix D documents the policy network, training settings, preference grids, augmentation, candidate retention, and the matched timing procedure; Appendix E.1 identifies the comparison methods. The core implementation code is provided in the Supplementary Material.

REFERENCES

Charles Audet, Jean Bigeon, Dominique Cartier, Sébastien Le Digabel, and Ludovic Salomon. Performance indicators in multiobjective optimization. *European Journal of Operational Research*, 292(2):397–422, 2021. doi: 10.1016/j.ejor.2020.11.016.

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

Jinbiao Chen, Jiahai Wang, Zizhen Zhang, Zhiguang Cao, Te Ye, and Siyuan Chen. Efficient meta neural heuristic for multi-objective combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, pp. 56825–56837, 2023a.

Jinbiao Chen, Zizhen Zhang, Zhiguang Cao, Yaoxin Wu, Yining Ma, Te Ye, and Jiahai Wang. Neural multi-objective combinatorial optimization with diversity enhancement. In *Advances in Neural Information Processing Systems*, volume 36, pp. 39176–39188, 2023b.

Jinbiao Chen, Zhiguang Cao, Jiahai Wang, Yaoxin Wu, Hanzhang Qin, Zizhen Zhang, and Yue-Jiao Gong. Rethinking neural multi-objective combinatorial optimization via neat weight embedding. In *The Thirteenth International Conference on Learning Representations*, 2025.

Weiyu Chen, Hisao Ishibuchi, and Ke Shang. Fast greedy subset selection from large candidate solution sets in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 26(4):750–764, 2022. doi: 10.1109/tevc.2021.3103386.

Indraneel Das and John E Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8(3):631–657, 1998. doi: 10.1137/s1052623496307510.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. doi: 10.1109/4235.996017.

Matthias Ehrgott. *Multicriteria optimization*. Springer, second edition, 2005. doi: 10.1007/3-540-27659-9.

Mingfeng Fan, Yaoxin Wu, Zhiguang Cao, Wen Song, Guillaume Sartoretti, Huan Liu, and Guohua Wu. Conditional neural heuristic for multiobjective vehicle routing problems. *IEEE Transactions on Neural Networks and Learning Systems*, 36(3):4677–4689, 2025a. doi: 10.1109/tnnls.2024.3371706.

Mingfeng Fan, Jianan Zhou, Yifeng Zhang, Yaoxin Wu, Jinbiao Chen, and Guillaume Adrien Sartoretti. Preference-driven multi-objective combinatorial optimization with conditional computation. In *Advances in Neural Information Processing Systems*, 2025b.

Xiaoyu Fu, Shenshen Gu, Chee-Meng Chew, and Tengfei Li. Towards generalizable meta-deep reinforcement learning algorithm for multiobjective traveling salesman problems. *IEEE Transactions on Artificial Intelligence*, 7(4):2238–2252, 2026. doi: 10.1109/tai.2025.3614210.

Hao Gao, Xin Xu, Changxin Zhang, and Xing Zhou. A bidirectional parameter transfer reinforcement learning approach for bi-objectives traveling salesman problem. In *2022 6th Asian Conference on Artificial Intelligence Technology (ACAIT)*, pp. 1–6. IEEE, 2022. doi: 10.1109/acait56212.2022.10137925.

Keld Helsgaun. An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, 2000. doi: 10.1016/S0377-2217(99)00284-2.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019.

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, pp. 21188–21198, 2020.

Fengxiaoxiao Li, Xiao Mao, Mingfeng Fan, Yifeng Zhang, Yi Li, Tanishq Duhan, and Guillaume Sartoretti. CAMO: A conditional neural solver for the multi-objective multiple traveling salesman problem. *arXiv preprint arXiv:2603.19074*, 2026.

Kaiwen Li, Tao Zhang, and Rui Wang. Deep reinforcement learning for multiobjective optimization. *IEEE Transactions on Cybernetics*, 51(6):3103–3114, 2021. doi: 10.1109/tcyb.2020.2977661.

Shicheng Li, Feng Wang, Qi He, and Xujie Wang. Deep reinforcement learning for multi-objective combinatorial optimization: A case study on multi-objective traveling salesman problem. *Swarm and Evolutionary Computation*, 83:101398, 2023. doi: 10.1016/j.swevo.2023.101398.

Xi Lin, Zhiyuan Yang, and Qingfu Zhang. Pareto set learning for neural multi-objective combinatorial optimization. In *International Conference on Learning Representations*, 2022.

Wei Liu, Yaoxin Wu, Yingqian Zhang, Thomas Bäck, and Yingjie Fan. Adversarial instance generation and robust training for neural combinatorial optimization with multiple objectives. *arXiv preprint arXiv:2601.01665*, 2026.

Yongfan Lu, Zixiang Di, Bingdong Li, Shengcai Liu, Hong Qian, Peng Yang, Ke Tang, and Aimin Zhou. Context-aware diversity enhancement for neural multi-objective combinatorial optimization. *arXiv preprint arXiv:2405.08604v3*, 2025. URL https://arxiv.org/abs/2405.08604v3.

Thibaut Lust and Jacques Teghem. The multiobjective multidimensional knapsack problem: a survey and a new approach. *International Transactions in Operational Research*, 19(4):495–520, 2012. doi: 10.1111/j.1475-3995.2011.00840.x.

Filip Rydin, Attila Lischka, Jiaming Wu, Morteza Haghir Chehreghani, and Balázs Kulcsár. Beyond simple graphs: Neural multi-objective routing on multigraphs. In *International Conference on Learning Representations*, 2026.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Jialong Shi, Qingfu Zhang, and Jianyong Sun. PPLS/D: Parallel Pareto local search based on decomposition. *IEEE Transactions on Cybernetics*, 50(3):1060–1071, 2020. doi: 10.1109/tcyb.2018.2880256.

Jialong Shi, Jianyong Sun, Qingfu Zhang, Haotian Zhang, and Ye Fan. Improving Pareto local search using cooperative parallelism strategies for multiobjective combinatorial optimization. *IEEE Transactions on Cybernetics*, 54(4):2369–2382, 2024. doi: 10.1109/TCYB.2022.3226744.

Dawei Su, Zizhen Zhang, Jinbiao Chen, and Zhanhong Fang. A discrete diffusion-based approach for solving multi-objective traveling salesman problem. In *2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 374–379. IEEE, 2024. doi: 10.1109/smc54092.2024.10831808.

Liting Wang, Chao Song, Yu Sun, Cuihua Lu, and Qinghua Chen. A Neural Multi-Objective Capacitated Vehicle Routing Optimization Algorithm Based on Preference Adjustment. *Electronics*, 12 (19):4167, 2023. doi: 10.3390/electronics12194167.

Zhenkun Wang, Shunyu Yao, Genghui Li, and Qingfu Zhang. Multiobjective combinatorial optimization using a single deep reinforcement learning model. *IEEE Transactions on Cybernetics*, 54(3):1984–1996, 2024. doi: 10.1109/tcyb.2023.3312476.

Huiqing Wei, Fei Han, Qing Liu, and Henry Han. RM-MOCO: A fast-solving model for neural multi-objective combinatorial optimization based on retention. *Complex System Modeling and Simulation*, 5(2):125–137, 2025. doi: 10.23919/csms.2024.0029.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992. doi: 10.1007/bf00992696.

Rixin Wu, Ran Wang, Jie Hao, Qiang Wu, Ping Wang, and Dusit Niyato. Multiobjective vehicle routing optimization with time windows: A hybrid approach using deep reinforcement learning and NSGA-II. *IEEE Transactions on Intelligent Transportation Systems*, 26(3):4032–4047, 2025. doi: 10.1109/tits.2024.3515997.

Yaoxin Wu, Mingfeng Fan, Zhiguang Cao, Ruobin Gao, Yaqing Hou, and Guillaume Sartoretti. Collaborative deep reinforcement learning for solving multi-objective vehicle routing problems. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pp. 1956–1965, 2024.

Te Ye, Zizhen Zhang, Jinbiao Chen, and Jiahai Wang. Weight-specific-decoder attention model to solve multiobjective combinatorial optimization problems. In *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2839–2844. IEEE, 2022. doi: 10.1109/smc53654.2022.9945568.

Sandra Zajac and Sandra Huber. Objectives and methods in multi-objective routing problems: a survey and classification scheme. *European Journal of Operational Research*, 290(1):1–25, 2021. doi: 10.1016/j.ejor.2020.07.005.

Haotian Zhang, Jialong Shi, Jianyong Sun, Qingfu Zhang, and Zongben Xu. Neuro-PLS: A generalizable local search framework for multiobjective combinatorial optimization. *IEEE Transactions on Evolutionary Computation*, 30(4):1564–1578, 2026. doi: 10.1109/tevc.2025.3589640.

Qingfu Zhang and Hui Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007. doi: 10.1109/tevc.2007.892759.

Yongxin Zhang, Jiahai Wang, Zizhen Zhang, and Yalan Zhou. MODRL/D-EL: Multiobjective deep reinforcement learning with evolutionary learning for multiobjective optimization. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2021. doi: 10.1109/ijcnn52387.2021.9534083.

Zizhen Zhang, Zhiyuan Wu, Hang Zhang, and Jiahai Wang. Meta-learning-based deep reinforcement learning for multiobjective optimization problems. *IEEE Transactions on Neural Networks and Learning Systems*, 34(10):7978–7991, 2023. doi: 10.1109/tnnls.2022.3148435.

Zhi Zheng, Shunyu Yao, Genghui Li, Linxi Han, and Zhenkun Wang. Pareto improver: Learning improvement heuristics for multi-objective route planning. *IEEE Transactions on Intelligent Transportation Systems*, 25(1):1033–1043, 2024. doi: 10.1109/tits.2023.3313688.

A Problem Definitions and Construction States

A.1 Notation and Objective Orientation

We follow the notation in the main paper: n is the problem size, B the training batch size, K the number of POMO candidates per instance, and Q the number of evaluation preferences. Indices b, k, ℓ, and t denote an instance, candidate, objective, and construction step, respectively. Preferences lie on Δ^{m-1}, and instances are sampled from the task-specific distributions below. Define minimization costs by $\mathbf{c}(x; I) = \mathbf{f}(x; I)$ for TSP and CVRP, and $\mathbf{c}(x; I) = -\mathbf{f}(x; I)$ for KP. The scalarized reward is consequently $R(x; I, \mathbf{w}) = -\mathbf{w}^\top \mathbf{c}(x; I)$ for every task. This scalarization uses the original task costs; the local normalization used for HV contributions does not replace the scalarized reward.

A.2 Bi- and Three-Objective Traveling Salesman Problems

Each of the n nodes has one two-dimensional coordinate $\mathbf{u}_i^{(\ell)}$ for each objective ℓ. Define $d_{ij}^{(\ell)} = \|\mathbf{u}_i^{(\ell)} - \mathbf{u}_j^{(\ell)}\|_2$. A feasible tour is a permutation $\sigma = (\sigma_1, \ldots, \sigma_n)$ with $\sigma_{n+1} = \sigma_1$. Its objective values are

$$f_\ell(\sigma; I) = \sum_{t=1}^{n} d_{\sigma_t, \sigma_{t+1}}^{(\ell)}, \qquad \ell = 1, \ldots, m, \tag{18}$$

and all objectives are minimized. Bi-TSP uses $m = 2$ and Tri-TSP uses $m = 3$. Coordinates are sampled independently and uniformly from $[0, 1]^{2m}$ for each node, following the benchmark construction used in neural MOCO (Lin et al., 2022; Chen et al., 2025). Both tasks use sizes $n \in \{20, 50, 100\}$.

The input feature of a node concatenates its m coordinate pairs. The construction state records the first node, current node, and visited set. An action selects an unvisited node; the tour closes to its first node after all n nodes have been visited. POMO candidate k starts from node k, with $K = n$. This initial choice is fixed rather than sampled. The decoder uses the first and current node embeddings, and masks visited nodes.

A.3 Bi-Objective Capacitated Vehicle Routing Problem

There are n customers, one depot indexed by 0, two-dimensional coordinates $\mathbf{u}_0, \ldots, \mathbf{u}_n$, customer demands $d_i > 0$, and vehicle capacity C_{v}. A solution $\mathcal{R}$ consists of routes that start and end at the depot. Each customer is served exactly once and the sum of demands on every route is at most C_{v}. For a route $\rho = (0, i_1, \ldots, i_h, 0)$, let $L(\rho)$ be the sum of Euclidean lengths of consecutive edges. The two objectives are

$$f_1(\mathcal{R}; I) = \sum_{\rho \in \mathcal{R}} L(\rho), \qquad f_2(\mathcal{R}; I) = \max_{\rho \in \mathcal{R}} L(\rho). \tag{19}$$

Both total distance and the longest route length (the makespan under unit travel speed) are minimized.

Depot and customer coordinates are independent uniform samples from $[0, 1]^2$. Demands are independent uniform integers in $\{1, \ldots, 9\}$. Capacities are 30, 40, and 50 at sizes 20, 50, and 100, respectively. The network and environment use normalized demands d_i / C_{v} and unit initial vehicle capacity.

The state contains the current node, visited customers, and remaining vehicle capacity. An action visits a feasible unserved customer or returns to the depot, where capacity is reset. Customers that have been served or exceed the remaining capacity are masked; the implementation uses a 10^{-5} tolerance for the normalized-capacity comparison. Consecutive depot visits are disallowed for unfinished trajectories. A trajectory finishes after all customers are served and the vehicle has returned to the depot. The depot and first customer are fixed initialization actions: POMO candidate k first visits customer k, with $K = n$. The decoder context combines the current-node embedding and remaining capacity. Customer features contain coordinates and normalized demand; depot coordinates have a separate input projection.

A.4 Bi-Objective Knapsack Problem

Item i has weight ω_i and two profits $v_i^{(1)}, v_i^{(2)}$. For binary decisions $x_i \in \{0,1\}$ and capacity C_{k}, the feasible set and objectives are

$$\sum_{i=1}^{n} \omega_i x_i \le C_{\mathrm{k}}, \qquad f_\ell(x; I) = \sum_{i=1}^{n} v_i^{(\ell)} x_i, \quad \ell \in \{1,2\}. \tag{20}$$

Both profits are maximized. Weights and profits are sampled independently and uniformly from $[0,1]$. Capacity is 12.5 for $n = 50$ and 25 for $n \in \{100, 200\}$. Negating profits converts this task to minimization costs for dominance and HV calculations.

An item feature is $(\omega_i, v_i^{(1)}, v_i^{(2)})$. The construction state records selected items, the last selected item, and remaining capacity. An action selects an unselected item that fits. Construction terminates when no remaining item fits. POMO candidate k starts with item k, with $K = n$; all items fit initially under these benchmark capacities. The current decoder query combines the last-item embedding and remaining capacity. Finished trajectories are padded with a dummy item of zero weight and zero profit until the batch finishes.

A.5 Trajectory Probability and Feasibility Masks

For candidate (b,k), $\tau_{b,k} = (a_1, \ldots, a_{T_{b,k}})$ is the action sequence up to its logical termination. The state s_t includes the instance and partial construction needed to determine feasible actions $\mathcal{A}(s_t)$. Given logits g_θ, a learned action has probability

$$\pi_\theta(a \mid s_t, \mathbf{w}_b) = \frac{\exp g_\theta(a; s_t, \mathbf{w}_b)}{\sum_{a' \in \mathcal{A}(s_t)} \exp g_\theta(a'; s_t, \mathbf{w}_b)}, \qquad a \in \mathcal{A}(s_t), \tag{21}$$

with probability zero outside $\mathcal{A}(s_t)$. Masking is applied before softmax. Environment transitions are deterministic given the instance and actions; replaying stored actions reconstructs the same feasibility states.

Let $\mathcal{T}_{b,k}^{\mathrm{learn}}$ denote the steps at which the action is chosen by the policy rather than by fixed POMO initialization. Conditional on the designated start,

$$\log \pi_\theta(\tau_{b,k} \mid I_b, \mathbf{w}_b) = \sum_{t \in \mathcal{T}_{b,k}^{\mathrm{learn}}} \log \pi_\theta(a_t \mid s_t, \mathbf{w}_b). \tag{22}$$

Fixed initialization actions have probability one and log-probability zero. KP dummy actions are excluded from both current and stored PPO log-probability sums. Finished CVRP trajectories remain at the depot with probability one. Thus padded execution steps do not change the complete-trajectory likelihood ratio.

B Pareto Concepts and Hypervolume

B.1 Dominance, Pareto Sets, and Finite Approximations

In minimization cost coordinates, $\mathbf{c}(x)$ dominates $\mathbf{c}(y)$, written $\mathbf{c}(x) \prec \mathbf{c}(y)$, if

$$c_\ell(x) \le c_\ell(y) \text{ for every } \ell, \qquad c_{\ell'}(x) < c_{\ell'}(y) \text{ for at least one } \ell'. \tag{23}$$

The Pareto set contains undominated feasible decisions, and the Pareto front is its image in objective space (Ehrgott, 2005). Pareto filtering retains undominated vectors from a finite generated set. Exact duplicates do not dominate one another, but one representative suffices for evaluation HV.

B.2 Evaluation Hypervolume

For a finite cost-vector set $\mathcal{F}$, a fixed reference point $\mathbf{r}$, and m-dimensional Lebesgue measure λ_m, define

$$\mathrm{HV}_{\mathbf{r}}(\mathcal{F}) = \lambda_m \left(\bigcup_{\mathbf{y} \in \mathcal{F}:\, \mathbf{y} \le \mathbf{r}} \prod_{\ell=1}^{m} [y_\ell, r_\ell] \right). \tag{24}$$

Only vectors weakly better than the reference in all coordinates define nonempty dominated boxes. The union counts overlapping regions once. Dominated vectors and exact duplicates do not change its volume, and $\mathrm{HV}_{\mathbf{r}}(\varnothing) = 0$.

Using fixed normalization bounds $\mathbf{z}$ and $\mathbf{r}$ with $r_\ell > z_\ell$, the reported metric is

$$\mathrm{nHV}_{\mathbf{r},\mathbf{z}}(\mathcal{F}) = \frac{\mathrm{HV}_{\mathbf{r}}(\mathcal{F})}{\prod_{\ell=1}^{m}(r_\ell - z_\ell)}. \tag{25}$$

Higher values indicate greater dominated volume (Audet et al., 2021). The same bounds are used for every compared method in a problem–size setting; they are not estimated separately from each method's outputs. Table 5 gives the bounds in minimization coordinates. In particular, KP profits and both metric bounds are sign-converted together.

Table 5: Fixed evaluation bounds. Slash-separated entries follow the listed sizes. These bounds are distinct from the candidate-dependent training normalization.

Problem	Size	Reference $\mathbf{r}$	Normalization bound $\mathbf{z}$
Bi-TSP	20 / 50 / 100	$(20, 20)$ / $(35, 35)$ / $(65, 65)$	$(0, 0)$
Bi-CVRP	20 / 50 / 100	$(30, 4)$ / $(45, 4)$ / $(80, 4)$	$(0, 0)$
Bi-KP	50	$(-5, -5)$	$(-30, -30)$
Bi-KP	100	$(-20, -20)$	$(-50, -50)$
Bi-KP	200	$(-30, -30)$	$(-75, -75)$
Tri-TSP	20	$(20, 20, 20)$	$(0, 0, 0)$
Tri-TSP	50	$(35, 35, 35)$	$(0, 0, 0)$
Tri-TSP	100	$(65, 65, 65)$	$(0, 0, 0)$

B.3 Local Training Hypervolume and Indexed Candidates

For one instance and one sampled preference, S_b is an indexed collection of K candidates. Using the local bounds and normalized improvement vectors from Equations (6)–(8), its training HV is

$$\mathrm{HV}(S_b) = \lambda_m\left(\bigcup_{k=1}^{K}\prod_{\ell=1}^{m}[0, v_{b,k,\ell}]\right). \tag{26}$$

For this fixed group geometry, the coordinate change divides cost-space HV by $\prod_\ell(r_{b,\ell} - z_{b,\ell})$. The same transformation is retained when candidate k is removed; recomputing extrema after removal would change the quantity being compared.

The difference $C_{b,k} = \mathrm{HV}(S_b) - \mathrm{HV}(S_b \setminus \{k\})$ removes one indexed candidate, not all candidates with the same objective vector. All dominated and duplicate candidates remain in this calculation: they can preserve coverage after another candidate is removed. Contributions are computed exactly up to floating-point precision for the finite group.

C Advantage Redistribution and Policy Updates

C.1 Batch Normalization and Exact Activation Conditions

Write $A^{\mathrm{raw}}_{b,k} = R_{b,k} - K^{-1}\sum_j R_{b,j}$. For a batch $\mathcal{B}$ with $N_{\mathcal{B}} = BK > 1$ entries, the normalization uses

$$\mu_{\mathcal{B}} = \frac{1}{N_{\mathcal{B}}}\sum_{b,k} A^{\mathrm{raw}}_{b,k}, \qquad s_{\mathcal{B}} = \left[\frac{\sum_{b,k}(A^{\mathrm{raw}}_{b,k} - \mu_{\mathcal{B}})^2}{N_{\mathcal{B}} - 1}\right]^{1/2}, \tag{27}$$

These statistics are used in Equation (4); $s_{\mathcal{B}}$ is the sample standard deviation across all instance–candidate entries, including the final partial batch of an epoch.

Set $p_{b,k} = C_{b,k} / \sum_j C_{b,j}$ if the sum is positive, and zero otherwise. Let $\mathcal{P}_b = \{k : A_{b,k} > 0\}$ and $M_b^+ = \sum_{k \in \mathcal{P}_b} A_{b,k}$. With $\varepsilon_A = 10^{-8}$, redistribution is activated only if all of the following hold:

$$\begin{aligned} |\mathcal{P}_b| > 1, \qquad & M_b^+ > \varepsilon_A, \\ \bar{p}_b^+ \text{ is finite}, \qquad & \bar{p}_b^+ > \varepsilon_A, \\ \max_{k \in \mathcal{P}_b} p_{b,k} - \min_{k \in \mathcal{P}_b} p_{b,k} > \varepsilon_A, & \end{aligned} \tag{28}$$

where $\bar{p}_b^+ = \sum_{k \in \mathcal{P}_b} p_{b,k} / \max(|\mathcal{P}_b|, 1)$. An empty positive group has mean zero and retains its original advantages. In the main paper, $\mathcal{G}$ denotes the set of groups satisfying these conditions.

Active groups use Equations (12)–(13). The weighted denominator is lower-bounded by ε_A numerically; this safeguard does not bind because $q_{b,k} \geq 1$ and $M_b^+ > \varepsilon_A$. Non-positive advantages and groups that fail any activation condition remain unchanged.

The contribution threshold applies to the positive group's mean, not to every candidate individually. Non-finite objective inputs are rejected before contributions are calculated.

C.2 Action Entropy and PPO Replay

Let $P^\theta_{b,k,t}(a)$ be the decoder distribution at a replayed state. If $\mathcal{T}_{\text{dec}}$ denotes the batch execution steps after fixed POMO initialization, the implemented entropy term is

$$\mathcal{H}(\pi_\theta) = \sum_{t \in \mathcal{T}_{\text{dec}}} \frac{1}{BK} \sum_{b=1}^{B} \sum_{k=1}^{K} \left[-\sum_a P^\theta_{b,k,t}(a) \log\big(P^\theta_{b,k,t}(a) + 10^{-10}\big) \right]. \tag{29}$$

This is the numerically stabilized action-distribution entropy: it is averaged over instances and POMO candidates and summed, not averaged, over decoder steps. Fixed initialization steps contribute no entropy. In the KP implementation, decoder entropy is averaged over the whole padded batch, including already finished entries; dummy actions are nevertheless excluded from the PPO likelihood ratio as specified in Appendix A.5.

The PPO ratio and loss follow Equations (15)–(17), without trajectory-length normalization. Each learned action's current log-probability is stabilized as $\log(\pi_\theta(a_t \mid s_t, \mathbf{w}_b) + 10^{-10})$, while the stored rollout uses $\log \pi_{\theta_{\text{old}}}(a_t \mid s_t, \mathbf{w}_b)$; the equations show the unstabilized ratio. Clipping applies to the complete-trajectory ratio, not individual action ratios. During the five full-batch updates, stored actions, instances, preferences, old log-probabilities, and advantages remain fixed and detached; current-policy probabilities and entropy are recomputed at each step.

C.3 REINFORCE Control and Ablation Interpretation

The REINFORCE control uses the same weighted-sum rewards and POMO group baseline (Williams, 1992; Kwon et al., 2020), but directly applies the raw advantages $A^{\text{raw}}_{b,k} = R_{b,k} - K^{-1} \sum_j R_{b,j}$. Its loss is

$$\mathcal{L}_{\text{REINFORCE}}(\theta) = -\frac{1}{BK} \sum_{b=1}^{B} \sum_{k=1}^{K} \text{sg}\big(A^{\text{raw}}_{b,k}\big) \log \pi_\theta(\tau_{b,k} \mid I_b, \mathbf{w}_b), \tag{30}$$

where sg denotes stop-gradient and the log-probability is conditional on the fixed POMO start. The implementation takes one optimizer step per newly sampled batch, without batch-wide advantage standardization, an entropy bonus, or a clipped likelihood ratio. Fixed initialization and post-termination padding have zero log-probability contributions.

The standard PPO control uses batch-standardized advantages, the entropy term in Equation (29), and five clipped updates per batch. In this comparison, HGI-RL uses the same PPO update settings and adds marginal-HV redistribution of positive advantages. The comparison with PPO measures the effect of this redistribution. All three training rules share PCAN, the default policy network detailed in Appendix D.1, as well as instance and preference distributions, number of sampled training instances, optimizer configuration, and inference budget.

D Default Policy Network and Evaluation Protocol

D.1 Preference-Conditioned Attention Network (PCAN)

PCAN is the default preference-conditioned attention network used to evaluate HGI-RL and the two training controls. It follows attention-based autoregressive construction (Kool et al., 2019) with POMO multi-start sampling and group-relative baselines (Kwon et al., 2020). This architecture is held fixed across the three training rules. A task-specific linear projection embeds the node or item features described in Appendix A. A two-layer multilayer perceptron (MLP) maps $\mathbf{w}$ through widths $m \to 128 \to 128$, with a rectified linear unit (ReLU) between its layers. For input embedding $\mathbf{h}_i^{\text{in}}$ and preference embedding $\mathbf{e}_w$, encoder input is

$$\mathbf{h}_i^0 = \text{LayerNorm}\big(\text{MLP}_{\text{in}}([\mathbf{h}_i^{\text{in}}; \mathbf{e}_w])\big)\,, \tag{31}$$

where $[\cdot;\cdot]$ denotes concatenation and MLP_{in} has widths $256 \to 256 \to 128$ with an intermediate ReLU. Six attention encoder layers use embedding dimension 128, eight heads, query/key/value dimension 16 per head, and feed-forward dimension 512. Decoder logits are tanh-clipped with constant 10.

Preferences enter through the encoder features. The decoder uses the task-specific contexts described in Appendix A, without separately injecting the preference vector into its query. Table 4 in the main text reports the application of HGI-RL to WE-CA and CNH.

D.2 Training Distribution and Evaluation Grid

Training uses one independently sampled $\text{Dirichlet}(\mathbf{1}_m)$ preference per instance, where $\mathbf{1}_m$ is the m-dimensional vector of ones. This is uniform on the simplex; endpoints are not inserted explicitly during training. Each model is trained for 200 epochs, with 100,000 newly generated instances per epoch and batch size 64, retaining the final partial batch. Adam uses learning rate 10^{-4} and weight decay 10^{-6}; the learning rate is multiplied by 0.1 at epoch 180.

The training-rule ablation in Table 3 uses epoch-200 checkpoints for REINFORCE, PPO, and HGI-RL.

For a positive integer resolution H, the evaluation preference set is the Das–Dennis simplex lattice (Das & Dennis, 1998)

$$\mathcal{W}_H = \left\{ \left(\frac{h_1}{H}, \ldots, \frac{h_m}{H} \right) : h_\ell \in \mathbb{Z}_{\geq 0},\ \sum_\ell h_\ell = H \right\}, \qquad Q = \binom{H+m-1}{m-1}. \tag{32}$$

We use $H = 100$ for two objectives ($Q = 101$) and $H = 13$ for three objectives ($Q = 105$). These evaluation grids include simplex endpoints.

D.3 Augmentation, Candidate Retention, and Reporting

Augmentation is applied only at inference. The eight distance-preserving transformations of a coordinate pair are

$$\begin{aligned} \mathcal{T} = \{&(x,y), (y,x), (x,1-y), (y,1-x),\\ &(1-x,y), (1-y,x), (1-x,1-y), (1-y,1-x)\}. \end{aligned} \tag{33}$$

For an m-objective TSP, they are applied independently to the m coordinate pairs, giving 8^m transformations: 64 for Bi-TSP and 512 for Tri-TSP. CVRP transforms depot and customer coordinates together and leaves demands unchanged, giving eight transformations. KP uses no augmentation.

Let G denote the augmentation factor, with $G = 1$ for unaugmented runs. For each preference, greedy decoding constructs KG candidates. It retains the candidate with the highest scalarized reward across starts and transformations. The Q retained objective vectors are then Pareto filtered. Consequently, each evaluated set has at most Q distinct objective vectors, while the raw construction budget is QKG trajectories per instance. HV is computed on these retained outputs rather than on the full pool of POMO candidates.

Every setting uses 200 fixed same-size test instances, shared across compared internal variants. Let $\mathcal{F}_b$ be the Pareto-filtered set of retained cost vectors for test instance b. A reported HV value is the arithmetic mean of the per-instance normalized HV values:

$$\overline{\mathrm{nHV}} = \frac{1}{200} \sum_{b=1}^{200} \mathrm{nHV}_{\mathbf{r},\mathbf{z}}(\mathcal{F}_b). \tag{34}$$

D.4 Matched Evaluation Timing

The matched timing comparison between PCAN+HGI-RL and WE-CA covers 20 configurations: sizes 20/50/100 of Bi-TSP, Bi-CVRP, and Tri-TSP, each with and without augmentation, and unaugmented Bi-KP50/100. Each run evaluates 200 instances on the same RTX 4090-equipped machine. Corresponding runs use identical test files, verified by file hashes. The full wall-clock measurement includes model and data loading, inference, and evaluation postprocessing. Runs execute sequentially and are measured once.

Both methods use an evaluation batch size of 200 without augmentation. With augmentation, the batch size is 100 for Bi-TSP100 and 100/50/20 for Tri-TSP20/50/100, respectively; the other augmented configurations use 200. PCAN+HGI-RL has a lower full evaluation time in all 20 matched configurations. Comparisons use the original seconds before display rounding, which can produce ties in the tables.

E Baseline Methods and Additional Network Comparisons

E.1 Comparison Methods

The non-neural comparisons include the decomposition-based MOEA/D (Zhang & Li, 2007), the dominance-based NSGA-II (Deb et al., 2002), and PPLS/D-C, the cooperative Pareto local search method of Shi et al. (2024). WS-LKH solves weighted-sum TSP subproblems using the Lin–Kernighan–Helsgaun heuristic (Helsgaun, 2000); WS-DP uses dynamic programming for weighted-sum knapsack subproblems.

The neural comparisons comprise DRL-MOA with parameter transfer (Li et al., 2021), the meta-learning methods MDRL (Zhang et al., 2023) and EMNH (Chen et al., 2023a), and the single-policy methods PMOCO (Lin et al., 2022), CNH (Fan et al., 2025a), and WE-CA (Chen et al., 2025). The main tables distinguish standard and augmented evaluation, and Table 6 summarizes their network characteristics.

E.2 Model Characteristics

Table 6: Method characteristics and approximate trainable parameter counts. Baseline counts are taken from Chen et al. (2025); multi-model counts cover the model collection, whereas single-model counts refer to one policy. PCAN is our default policy network. HGI-RL adds no trainable parameters to the selected architecture.

Method	Main mechanism	Model type	Parameters
DRL-MOA	Transfer learning	Multi-model	133.37M
MDRL	Meta-learning	Multi-model	133.37M
EMNH	Meta-learning	Multi-model	133.37M
PMOCO	Hypernetwork	Single-model	1.50M
CNH	Size-aware decoder	Single-model	1.63M
WE-CA	Conditional attention	Single-model	1.47M
PCAN	Preference-conditioned encoder	Single-model	1.38M