
RoME: Domain-Robust Mixture-of-Experts for MILP Solution Prediction across Domains

Tianle Pu¹, Zijie Geng², Haoyang Liu², Shixuan Liu^{3,1}, Jie Wang², Li Zeng¹,
Chao Chen^{1*}, Changjun Fan^{1*}

¹Laboratory for Big Data and Decision, College of Systems Engineering,
National University of Defense Technology

²MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition,
University of Science and Technology of China

³College of Computer Science and Technology, National University of Defense Technology
{putl22, liushixuan, zengli24, chenc1997, fanchangjun}@nudt.edu.cn
{zijiegeng, dgyoung}@mail.ustc.edu.cn, jiewangx@ustc.edu.cn

Abstract

Mixed-Integer Linear Programming (MILP) is a fundamental and powerful framework for modeling complex optimization problems across diverse domains. Recently, learning-based methods have shown great promise in accelerating MILP solvers by predicting high-quality solutions. However, most existing approaches are developed and evaluated in single-domain settings, limiting their ability to generalize to unseen problem distributions. This limitation poses a major obstacle to building scalable and general-purpose learning-based solvers. To address this challenge, we introduce **RoME**, a domain-**R**obust **M**ixture-of-**E**xperts framework for predicting MILP solutions across domains. RoME dynamically routes problem instances to specialized experts based on learned task embeddings. The model is trained using a two-level distributionally robust optimization strategy: inter-domain to mitigate global shifts across domains, and intra-domain to enhance local robustness by introducing perturbations on task embeddings. We reveal that cross-domain training not only enhances the model’s generalization capability to unseen domains but also improves performance within each individual domain by encouraging the model to capture more general intrinsic combinatorial patterns. Specifically, a single RoME model trained on three domains achieves an average improvement of 67.7% then evaluated on five diverse domains. We further test the pretrained model on MIPLIB in a zero-shot setting, demonstrating its ability to deliver measurable performance gains on challenging real-world instances where existing learning-based approaches often struggle to generalize.

1 Introduction

Mixed-Integer Linear Programming (MILP) is a fundamental and expressive framework for modeling a wide range of real-world optimization problems, including logistics optimization [1], network design [2], scheduling [3], and industrial planning [4]. Due to its strong modeling capacity, MILP has become a cornerstone of operation research and combinatorial optimization [5–7]. However, MILPs are well known to be NP-hard, and solving large-scale instances with thousands of variables and constraints remains computationally intensive. Even state-of-the-art solvers such as Gurobi [8] and SCIP [9], which integrate advanced techniques like branch-and-bound, cutting planes, and heuristics, can struggle to deliver timely and scalable solutions as problem complexity grows.

*Corresponding author.

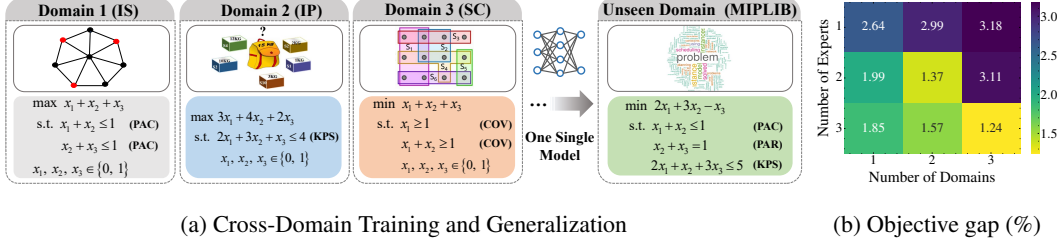


Figure 1: **Illustration of cross-domain training.** (a) A single model is trained on a collection of MILP domains with varying constraint types [10], such as PAC (set packing), KPS (knapsack), COV (set covering), and PAR (set partitioning), and evaluated on unseen domains such as MIPLIB. (b) We conduct experiments across five distinct domains, reporting the average objective gap in percentage relative to the best-known solution while varying the number of experts and training domains. Each value in the heatmap denotes the average percentage gap to the best-known solution. The results show that training on more diverse domains and selecting an appropriate number of experts can effectively reduce this objective gap.

To alleviate these limitations, recent research has explored the use of machine learning to accelerate MILP solving by identifying data-driven patterns in problem instances [11–14]. These approaches fall broadly into two categories. The first integrates neural networks into solver internals by replacing key components such as variable selection [11], cutting plane selection [12], and neighborhood destruction strategies in large neighborhood search [15, 16]. The second line of research, which we focus on in this paper, aims to directly predict initial feasible solutions [17–19], which are then refined by solvers to improve solution quality. These methods have shown promising results on large-scale MILPs and are gaining increasing attention. However, while this solution-prediction paradigm has shown strong empirical performance, most existing methods are trained and evaluated within single-domain settings, which presents two major challenges. First, the learned models often overfit to domain-specific patterns rather than capturing generalizable combinatorial structures, leading to limited generalization to unseen domains. Second, deploying such models in practice often requires collecting new training data for each specific scenario, limiting scalability and broader applications.

To build scalable learning-based solvers that generalize across domains, we advocate for a paradigm shift from single-domain training to frameworks that explicitly embrace cross-domain structural diversity. An illustration of the cross-domain training task is in Figure 1. A central insight of this work is that training on multiple MILP domains not only improves generalization under distribution shift but also enhances solution quality within each individual domain (see Section 4.2). Rather than treating distributional differences as noise, we view them as valuable signals that help the model capture transferable combinatorial patterns. In this light, cross-domain training naturally acts as a form of domain-level regularization, steering the model away from memorization and towards learning general principles that transfer across tasks.

In this paper, we propose RoME, a Domain-**R**obust **M**ixture-of-**E**xperts framework for MILP solution prediction. RoME aims to generalize effectively across various problem distributions while maintaining strong performance on specific tasks. The framework integrates a Mixture-of-Experts (MoE) architecture with a two-level Distributionally Robust Optimization (DRO) objective into a unified trainable framework. The MoE consists of multiple expert networks, each specializing in distinct MILP distributions. For each instance, the MoE dynamically routes it to specialized experts based on its graph embedding, allowing the model to adapt both its internal representations and final outputs to the unique structural characteristics of each instance. To further enhance robustness, RoME is trained using a two-level distributionally robust optimization strategy. (1) Inter-domain robustness is achieved through group-DRO, which minimizes the worst-case loss across domains, thereby reducing the impact of global distributional shifts. (2) Intra-domain robustness is ensured by applying isotropic perturbations to the task embeddings, which stabilizes expert selection and predictions.

To demonstrate the effectiveness of RoME, we conduct comprehensive experiments to demonstrate the effectiveness of RoME. Specifically, we train a single RoME model on three domains and evaluate it across five diverse domains, where it achieves an average improvement of 67.7% over strong baselines. We further test the pretrained model on MIPLIB in a zero-shot setting, demonstrating a significant performance gain on challenging instances that existing learning-based approaches often struggle to generalize. Finally, to gain a deeper understanding of RoME, we conduct interpretability

analyses that find expert activation patterns and task embeddings, shedding light on how the model captures and transfers structural knowledge across domains.

2 Related Work

2.1 Machine Learning for MILPs

Machine learning has been widely applied to the field of MILPs [20–23]. Recent work on learning-based MILP solvers can be roughly classified into two categories. The first line of research integrates machine learning to enhance the solving efficiency of conventional solvers. Researchers utilize a learned model to replace some key modules in the solvers, including variable selection [11, 24, 13], node selection [25–27], cutting plane selection [12, 28] and large neighborhood search [15, 16, 29–31]. Another line of research focuses on leveraging learning-based models to predict an initial solution for MILPs. Nair et al. [17] introduces the first solution-prediction framework for MILPs. Building on this foundation, Han et al. [18] proposes the Predict-and-Search (PS) framework, which incorporates a trust-region search to improve feasibility and solution quality. To further enhance prediction accuracy, Huang et al. [19] employs contrastive learning to train the solution-prediction network. Liu et al. [32] adopts a prediction–correction paradigm to obtain higher-quality solutions by iteratively refining initially mis-predicted variable assignments. Notably, Geng et al. [33] is the first to propose an unsupervised method that incorporates gradient information into the prediction model for MILPs, thereby significantly reducing the cost of collecting high-quality training data.

2.2 Cross-Domain Training Techniques

Mixture of Experts (MoE) [34] and Distributionally Robust Optimization (DRO) [35, 36] are two core methodologies for cross-domain training, where MoE gives the model with the ability to perceive and adapt to different domains, and DRO enhances robustness by balancing training across domains. Early studies [37] focus on designing dedicated experts for individual domains, while recent research [38, 39] has shifted toward sparse architectural variants. The MoE paradigm has been successfully applied to graph-based combinatorial optimization problems. For instance, MVMoE [40] tackles a suite of vehicle-routing variants, and MoE-style encoder-decoder frameworks, e.g., MAB-MTL [41], GCNCO [42], GOAL [43], consistently adopt a “header-encoder-decoder” architecture. In contrast, DRO guarantees robustness to potential domain shifts by minimizing the worst-case risk over an uncertainty set. Sagawa et al. [44] first introduces Group DRO, which balances the losses across different groups. CCD-DG [45] builds a class-conditioned Wasserstein ball and automatically tunes its radius to protect against conditional shifts, while Moderately-DRO [46] and its stochastic variant [47] improves exploration of unseen domains and offers tighter generalization bounds with lower sampling complexity.

3 Methodology

In this section, we present **RoME**, a domain-**R**obust **M**ixture-of-**E**xperts framework for cross-domain MILP prediction. This section is organized as follows. Section 3.1 introduces the problem formulation, the predict-and-search (PS) framework, and the cross-domain learning settings. Section 3.2 presents the Mixture-of-Experts (MoE) architecture for instance-adaptive structure-awareness across different domains. Section 3.3 presents a robust training objective that effectively improves the robustness of the MoE model training. Finally, Section 3.4 introduces a group-level Distributionally Robust Optimization (DRO) scheme to handle the cross-domain training process.

3.1 Problem Formulation

We focus on the task of learning to predict high-quality solutions for Mixed-Integer Linear Programming (MILP) problems. A MILP instance $\mathcal{I}$ is defined as:

$$\min_{\mathbf{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}} \{ \mathbf{c}^\top \mathbf{x} \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u} \}, \quad (1)$$

where $\mathbf{x} \in \mathbb{R}^n$ denotes the decision variables, with the first p entries being integer and the remaining $n - p$ continuous. The vector $\mathbf{c} \in \mathbb{R}^n$ denotes the coefficients of the linear objective, the constraints

are defined by the matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ and the right-hand side vector $\mathbf{b} \in \mathbb{R}^m$, and the variable bounds are given by $\mathbf{l} \in (\mathbb{R} \cup \{-\infty\})^n$ and $\mathbf{u} = (\mathbb{R} \cup \{+\infty\})^n$. Without loss of generality, we focus on binary integer variables, and general integer variables can be handled via standard preprocessing techniques [17].

We encode each MILP instance as a bipartite graph $\mathcal{G} = (\mathcal{W} \cup \mathcal{V}, \mathcal{E})$, where $\mathcal{W}$ and $\mathcal{V}$ denote the sets of constraint and variable nodes, respectively, and the edge set $\mathcal{E}$ corresponds to non-zero entries in $\mathbf{A}$. Each node and edge is associated with a set of features derived from problem coefficients and structural attributes. Such a bipartite graph can completely describe a MILP instance, enabling graph neural networks (GNNs) to process the instances and predict their solutions.

We adopt the Predict-and-Search (PS) paradigm to approximate the solution distribution of a given MILP. Specifically, the distribution is defined via an energy function that assigns lower energy to high-quality feasible solutions and infinite energy to infeasible ones:

$$p(\mathbf{x} \mid \mathcal{I}) = \frac{\exp(-E(\mathbf{x} \mid \mathcal{I}))}{\sum_{\mathbf{x}'} \exp(-E(\mathbf{x}' \mid \mathcal{I}))}, \quad \text{where} \quad E(\mathbf{x} \mid \mathcal{I}) = \begin{cases} \mathbf{c}^\top \mathbf{x}, & \text{if } \mathbf{x} \text{ is feasible,} \\ +\infty, & \text{otherwise.} \end{cases} \quad (2)$$

Our goal is to learn distribution $p_\theta(\mathbf{x} \mid \mathcal{I})$ for a given instance $\mathcal{I}$. To make learning tractable, we assume a fully factorized solution distribution over the binary variables, i.e., $p_\theta(\mathbf{x} \mid \mathcal{I}) = \prod_{i=1}^p p_\theta(x_i \mid \mathcal{I})$, where $p_\theta(x_i \mid \mathcal{I})$ denotes the predicted marginal probability for variable x_i . To do so, we use a GNN model to output a p -dimension vector $\hat{\mathbf{x}} = \mathbf{f}_\theta(\mathcal{I}) = (\hat{x}_1, \dots, \hat{x}_p)^\top \in [0, 1]^p$, where $\hat{x}_j = p_\theta(x_j = 1 \mid \mathcal{I})$. To train the model, we use a weighted set of feasible solutions $\{\mathbf{x}^{(i)}\}_{i=1}^N$ as supervised signals, where each solution is assigned a weight $w_i \propto \exp(-\mathbf{c}^\top \mathbf{x}^{(i)})$. We let $x_j^{(i)} = p_\theta(x_j^{(i)} = 1 \mid \mathcal{I})$. Then, the training loss function is a binary cross-entropy loss defined as:

$$\mathcal{L}_{\text{BCE}}(\theta \mid \mathcal{I}) = - \sum_{i=1}^N \sum_{j=1}^p w_i \cdot \left[x_j^{(i)} \log \hat{x}_j + (1 - x_j^{(i)}) \log(1 - \hat{x}_j) \right]. \quad (3)$$

At inference time, the GNN model outputs a predicted marginals $\hat{\mathbf{x}} \in [0, 1]^p$. A standard MILP solver, e.g., Gurobi or SCIP is then used to search for a feasible solution in a local neighborhood around $\hat{\mathbf{x}}$ by solving the following trust region problem:

$$\min_{\mathbf{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}} \{ \mathbf{c}^\top \mathbf{x} \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \mathbf{x}_{1:p} \in \mathcal{B}(\hat{\mathbf{x}}, \Delta) \}, \quad (4)$$

where the trust region $\mathcal{B}(\hat{\mathbf{x}}, \Delta) := \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}_{1:p} - \hat{\mathbf{x}}\|_1 \leq \Delta\}$ constrains the solver to remain close to the predicted binary configuration.

It's evident that the quality of the final solution depends heavily on the accuracy of the predicted marginals. Poor predictions can mislead the solver and limit optimization performance. Therefore, we focus on training a predictive model $\mathbf{f}_\theta$ that can accurately predict the marginal probabilities.

3.2 Mixture-of-Experts with Structure-Aware Routing

We consider the setting where training samples are drawn from K different domains $\{\mathcal{D}_1, \dots, \mathcal{D}_K\}$, each corresponding to a structurally distinct MILP family. Our goal is to learn a single unified model that can robustly generalize not only across these diverse domains, but also those domains unseen during training. However, due to the significant combinatorial differences between domains, capturing all structural variations within a single monolithic model is challenging.

To address this challenge, we adopt a Mixture-of-Experts (MoE) framework that comprises three key components: a shared graph encoder, multiple expert networks, and a task decoder. The shared graph encoder learns common structural representations across tasks, while the expert networks capture task-specific decision patterns under the guidance of a dynamic gating mechanism that adaptively assigns expert weights based on task-level embeddings. Finally, the task decoder integrates the aggregated expert outputs to generate the final prediction. This design enables structure-aware representation learning and output specialization.

Shared Graph Encoder Given a MILP instance $\mathcal{I}$, represented as a bipartite graph $\mathcal{G}$, we apply a graph neural network (GNN) to extract node embeddings $\{\mathbf{h}_1, \dots, \mathbf{h}_p\} \subset \mathbb{R}^d$, where d is the dimension of the hidden space. We then compute a global graph embedding $\mathbf{h}_\mathcal{G} \in \mathbb{R}^d$ by mean pooling over variable nodes, i.e., $\mathbf{h}_\mathcal{G} = \frac{1}{p} \sum_{j=1}^p \mathbf{h}_j$. In this work, we follow the GNN encoder design used in [18].

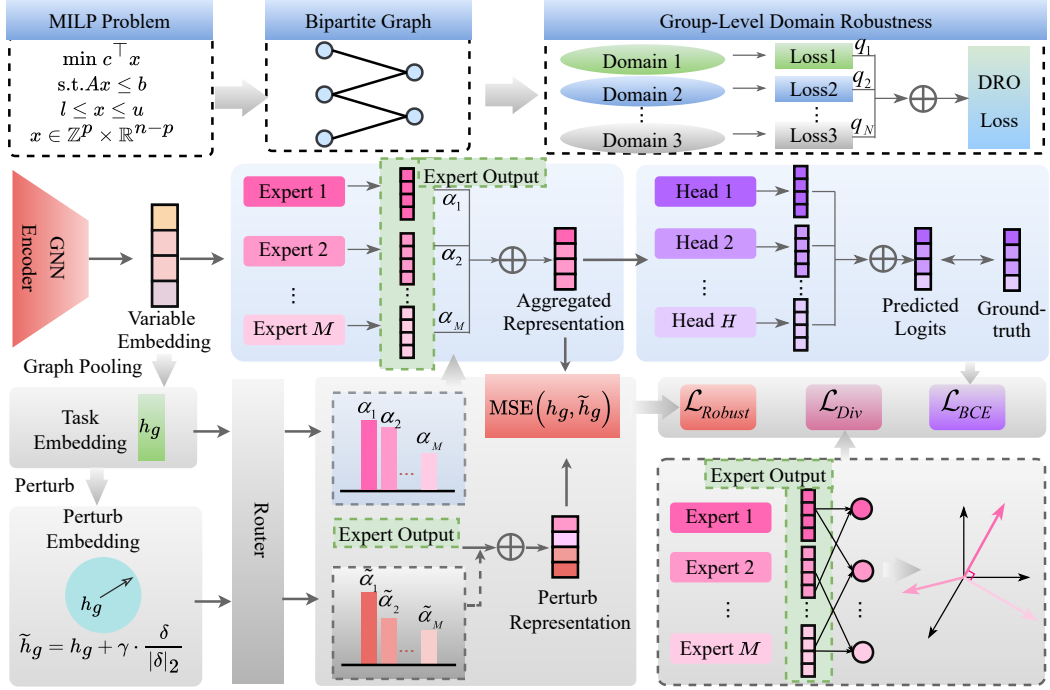


Figure 2: **The overview of RoME.** RoME employ a MoE architecture with structure-aware representation learning. For cross-domain training, RoME proposes a robust training object containing binary cross-entropy loss $\mathcal{L}_{BCE}$, expert diversity regularization $\mathcal{L}_{Div}$ and fuzzy membership consistency loss $\mathcal{L}_{Robust}$. To further enhance generalization across structurally diverse MILP families, RoME uses a group-level domain robust scheme to construct DRO loss.

Multiple Expert Network To capture structural diversity, we define a set of M expert networks $\{\mathbf{f}_{\theta}^{(1)}, \dots, \mathbf{f}_{\theta}^{(M)}\}$, where each expert $\mathbf{f}_{\theta}^{(m)} : \mathbb{R}^d \rightarrow \mathbb{R}^d$ maps the variable embeddings into an expert-specific representation space:

$$\mathbf{z}_j^{(m)} = \mathbf{f}_{\theta}^{(m)}(\mathbf{h}_j). \quad (5)$$

Each expert can be interpreted as modeling a different mode of structural regularity, allowing the model to specialize across domains. To route instances to experts, we use a gating network $\mathbf{g}_{\theta}^{\text{enc}} : \mathbb{R}^d \rightarrow \mathbb{R}^M$ that maps the graph-level embedding $\mathbf{h}_G$ into a soft weighting over all experts:

$$\boldsymbol{\alpha} = \text{Softmax}(\mathbf{g}_{\theta}^{\text{enc}}(\mathbf{h}_G)/\tau), \quad (6)$$

where τ is a temperature parameter controlling the sharpness of the routing distribution. The vector $\boldsymbol{\alpha} \in [0, 1]^M$ can be interpreted as softly indicating which experts are most relevant for the current instance. The final representation for each variable is computed by aggregating the expert outputs:

$$\mathbf{z}_j = \sum_{m=1}^M \alpha_m \cdot \mathbf{f}_{\theta}^{(m)}(\mathbf{h}_j) \quad (7)$$

Task Decoder To further enhance specialization, we adopt a multi-head decoder that enables the model to learn and combine multiple task-specific representations. Specifically, we use the decoder set $\{d_{\theta}^{(1)}, \dots, d_{\theta}^{(H)}\}$ with H decoders, each mapping $\mathbf{z}_j$ to a scalar logit:

$$\sigma_j^{(h)} = d_{\theta}^{(h)}(\mathbf{z}_j). \quad (8)$$

Another gating network $\mathbf{g}_{\theta}^{\text{dec}}$ —which shares a similar architecture but different parameters with the encoder gate—produces a soft weighting vector $\boldsymbol{\beta} \in [0, 1]^H$ over the decoders:

$$\boldsymbol{\beta} = \text{Softmax}(\mathbf{g}_{\theta}^{\text{dec}}(\mathbf{h}_G)/\tau'), \quad (9)$$

where τ' is a temperature parameter controlling the sharpness of the routing distribution. The final logit is obtained by aggregating the outputs from all decoder heads:

$$\sigma_j = \sum_{h=1}^H \beta_h \cdot \sigma_j^{(h)}, \quad (10)$$

and the predicted marginal is given by

$$\hat{x}_j = \text{Sigmoid}(\sigma_j). \quad (11)$$

This flexible MoE design allows the model to dynamically adjust both internal representations and final outputs to match the structural characteristics of each instance. The expert network captures variable-level diversity, while the task decoder adds output-level specialization to further enhance generalization across domains.

3.3 Intra-Domain Robust Training Objective

To enable intra-domain robustness across structurally diverse instances in a domain, we introduce a composite loss function consisting of three components: a binary cross-entropy loss to supervise the training and a regularization term to promote output stability under domain uncertainty. Formally, for a given training instance $\mathcal{I}$, the total objective is given by:

$$\mathcal{L}(\theta|\mathcal{I}) = \mathcal{L}_{\text{BCE}} + \lambda_{\text{Div}} \cdot \mathcal{L}_{\text{Div}} + \lambda_{\text{Robust}} \cdot \mathcal{L}_{\text{Robust}}, \quad (12)$$

where λ_{Div} and λ_{Robust} are hyperparameters balancing these terms.

Binary Cross-Entropy Loss $\mathcal{L}_{\text{BCE}}$. As introduced in Section 3.1, we train the model by minimizing the weighted binary cross-entropy (3) between the predicted marginals $p_\theta(x_j = 1 | \mathcal{I})$ and a weighted set of feasible solutions sampled from the MILP instance. This loss serves as the primary signal for learning meaningful solution distributions.

Expert Diversity Regularization $\mathcal{L}_{\text{Div}}$. To encourage each expert to learn distinct behaviors and avoid mode collapse, we regularize the similarity among expert outputs. Recall that $\mathbf{z}_j^{(m)}$ denote the output embedding of expert m for variable j after the expert network layer. Let $\mathbf{z}^{(m)}$ denote the concatenation of the representation of all variables, i.e., $\mathbf{z}^{(m)} = \text{Concat}(\mathbf{z}_1^{(m)}, \dots, \mathbf{z}_p^{(m)})$. The diversity loss is then computed as:

$$\mathcal{L}_{\text{Div}} = \frac{1}{M(M-1)} \sum_{m \neq m'} \frac{|\langle \mathbf{z}^{(m)}, \mathbf{z}^{(m')} \rangle|}{\|\mathbf{z}^{(m)}\| \cdot \|\mathbf{z}^{(m')}\|}. \quad (13)$$

This term encourages orthogonal expert representations, promoting more diverse perspectives across MILP structures.

Fuzzy Membership Consistency Loss $\mathcal{L}_{\text{Robust}}$. To improve the model’s robustness to domain uncertainty, we adopt a fuzzy membership view of the expert routing mechanism. The routing vector $\alpha = \mathbf{g}_\theta(\mathbf{h}_\mathcal{G}) \in [0, 1]^M$, derived from the task embedding $\mathbf{h}_\mathcal{G}$, can be interpreted as a soft domain membership, indicating the degree to which each expert should be activated. However, for unseen or ambiguous MILP instances, the true domain identity may be uncertain or poorly represented in the training data. In such cases, small variations in the task embedding may lead to unstable routing decisions and inconsistent predictions. To address this, we encourage the model to maintain output consistency under slight changes in the domain assignment. Concretely, we apply an isotropic perturbation to the task embedding:

$$\tilde{\mathbf{h}}_\mathcal{G} = \mathbf{h}_\mathcal{G} + r \cdot \frac{\boldsymbol{\delta}}{\|\boldsymbol{\delta}\|_2}, \quad \boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad (14)$$

where r is a learnable scalar controlling the perturbation magnitude. This perturbed task embedding induces a new routing vector $\tilde{\alpha} = \mathbf{g}_\theta(\tilde{\mathbf{h}}_\mathcal{G})$, which results in a different mixture of expert representations $\tilde{\mathbf{z}}_j$. We then enforce consistency between the original and perturbed representations using the following loss:

$$\mathcal{L}_{\text{Robust}} = \frac{1}{p} \sum_{j=1}^p \|\tilde{\mathbf{z}}_j - \mathbf{z}_j\|_2^2. \quad (15)$$

This regularization encourages the model to produce stable representations even under uncertainty in domain assignment, thus improving its robustness to both domain shift and embedding noise. Beyond its regularization role, this loss is also theoretically motivated from two complementary perspectives. First, from a computational-complexity viewpoint, most MILP classes are NP-complete, implying that different MILPs can, in principle, be reduced to one another, a fact confirmed by recent advances in graph-based combinatorial optimization [5, 48]. Building on this insight, we exploit structural similarities among domains so that related problems route to overlapping expert subsets, enhancing cross-domain generalization. Perturbing task embeddings partially simulates such cross-domain shifts, enforcing the same expert outputs for perturbed and original embeddings strengthens this alignment and encourages domain-invariant reasoning. Second, from an adversarial-training viewpoint, embedding perturbations stabilize the training, helping RoME capture high-level patterns that persist across domains.

3.4 Inter-Domain Group-Level Robustness

To further enhance generalization across structurally diverse MILP families, we introduce a domain-level training strategy based on distributionally robust optimization (DRO). Rather than minimizing a uniform average of per-domain losses, we adopt a min-max objective that optimizes performance under the worst-case distribution over training domains. Formally, let $\mathcal{L}_k$ denote the expected loss on domain $\mathcal{D}_k$, estimated empirically via mini-batch sampling:

$$\mathcal{L}_k(\boldsymbol{\theta}) := \mathbb{E}_{\mathcal{I} \sim \mathcal{D}_k} [\mathcal{L}(\boldsymbol{\theta}|\mathcal{I})], \quad (16)$$

where $\mathcal{L}(\boldsymbol{\theta}|\mathcal{I})$ is the instance-level training loss as defined in Section 3.3. We introduce a probability vector $\mathbf{q} \in \Delta_K$, where Δ_K denotes the probability simplex over the K domains, satisfying $\sum_{k=1}^K q_k = 1$ and $q_k \geq 0$. We aim to optimize the following objective:

$$\min_{\boldsymbol{\theta}} \sup_{\mathbf{q} \in \Delta_K} \sum_{k=1}^K q_k \cdot \mathcal{L}_k(\boldsymbol{\theta}), \quad (17)$$

which seeks to minimize the worst-case expected loss under all convex combinations of domains-specific losses. To achieve this, we alternatively update $\boldsymbol{\theta}$ and $\mathbf{q}$. Intuitively, we maintain a distribution $\mathbf{q}$ over groups, with high masses on high-loss groups, and update on each example proportionally to the mass on its group. In each iteration, we first estimate $\mathcal{L}_k$ for a randomly chosen domain using a minibatch. Then, we update the domain weights $\mathbf{q}$ using exponential gradient ascent so that $q_k \propto \exp(\eta \cdot \mathcal{L}_k)$, where $\eta > 0$ is a hyperparameter controlling the sensitivity to high-loss domains. The resulting weights $\mathbf{q}$ are then normalized to form a valid probability vector over domains. Therefore, domains with higher empirical losses are thus assigned larger weights, allowing the model to focus learning capacity on harder or underrepresented domains. By coupling this domain-level reweighting with the instance-level regularization objectives introduced in Section 3.3, our training pipeline enables scalable and principled optimization for robust multi-domain MILP learning.

4 Experiments

We conduct comprehensive experiments to evaluate the effectiveness of RoME in learning a unified model for solving MILP problems from several different domains, including those in-distribution and out-of-distribution datasets, and zero-shot generalization to real-world instances from MIPLIB. We then conduct experiments to analyze the contributions of different components, as well as the featured patterns of our learned model. All experiments are conducted on a single machine with NVIDIA GeForce RTX 3090 GPUs and 24-core AMD EPYC 7402 CPUs (2.80 GHz). Code is available at <https://github.com/happypu326/RoME>. More implementation details can be found in Appendix A.

4.1 Experimental Setup

Datasets We conduct experiments on five MILP problem families widely studied in the literature: Independent Set (IS), Set Covering (SC), Item Placement (IP), Combinatorial Auctions (CA), and Workload Appointment (WA). The instances in the dataset have diverse kinds of combinatorial structures and constraints, with different levels of constraint coefficient sparsity, variable numbers, and objective complexity. For IS SC and CA, we follow the conventional instances generation process in existing works [11, 12, 18, 19]. The IP and WA datasets are from NeurIPS 2021 ML4CO competition [49]. Please see Appendix C for more detailed information on the benchmarks.

Table 1: Performance comparison across various MILP domains, under a 1,000s time limit. We train RoME on IS, IP and SC, while the performance of RoME on WA and CA is zero-shot performance. ‘↑’ indicates that higher is better, and ‘↓’ indicates that lower is better. We mark the **best values** in bold. We also report the improvement of our method over the traditional solvers in terms of gap_{abs} .

	IS (BKS 685.00)		IP (BKS 11.16)		SC (BKS 124.64)		WA (BKS 703.05)		CA (BKS 97524.37)	
	Obj ↑	Time ↓	Obj ↓	gap_{abs} ↓	Obj ↓	gap_{abs} ↓	Obj ↓	gap_{abs} ↓	Obj ↑	gap_{abs} ↓
Gurobi	685.00	57.35	11.43	0.27	125.21	0.57	703.47	0.42	97308.93	215.44
PS+Gurobi	685.00	14.67	11.40	0.24	125.17	0.53	703.47	0.42	97280.70	243.67
ConPS+Gurobi	685.00	20.13	11.36	0.20	125.18	0.54	703.47	0.42	97395.83	128.54
RoME+Gurobi	685.00	2.08	11.22	0.06	124.69	0.05	703.11	0.06	97489.28	35.09
Improvement	27.5 ×		77.8%		91.2%		85.7%		83.7%	

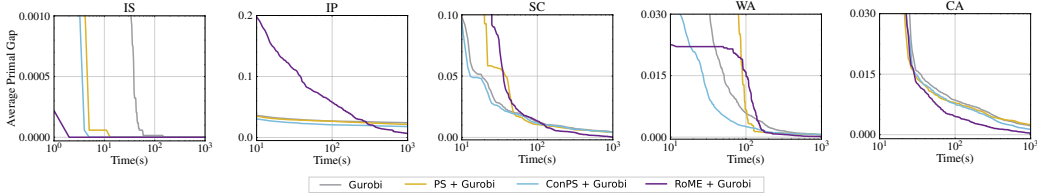


Figure 3: **The average primal gap of different methods over 100 instances as the solving process proceeds.** We use Gurobi for implementation and set the time limit to be 1,000 seconds.

Baselines We mainly consider two baselines, Predict-and-Search (PS) [18] and Contrastive Predict-and-Search (ConPS) [19]. Contrastive Predict-and-Search (ConPS) is a stronger baseline, leveraging contrastive learning to enhance the performance of PS. For ConPS, we set the ratio of positive to negative samples at ten, using low-quality solutions as negative samples. Our method and baselines can be integrated with conventional solvers such as SCIP [9] and Gurobi [8]. Therefore, we also include SCIP and Gurobi as baselines for a comprehensive comparison. Following Han et al. [18], Gurobi and SCIP are set to focus on finding better primal solutions.

Metrics We compare the performance of our method and baselines using the best objective value OBJ within 1,000 seconds. Following the setting in Han et al. [18], we run a single-thread Gurobi for 3,600 seconds and record the best objective value as the best-known solution (BKS) to approximate the optimal value. We calculate the absolute primal gap as the difference between the best objective found by the solvers and the BKS, i.e., $\text{gap}_{\text{abs}} := |\text{OBJ} - \text{BKS}|$. Within the same solving time, a lower absolute primal gap indicates stronger performance. For IS, where most test cases can be solved within the 1,000s, we additionally report the time cost to obtain the optimal value, with a lower time cost indicating a better performance.

Training and Evaluation Throughout all the performance experiments, we use *one unified model* for RoME. To train this model, we use a training dataset composed of 720 instances, where the dataset contains 240 IS instances, 240 IP instances and 240 SC instances. The validation set contains 240 instances with 80 IS, IP and SC instances, respectively. The performance of RoME on WA, CA and MIPLIB is the performance of zero-shot generalization. The baselines are trained individually on each dataset using 240 instances for training and 80 instances for validation. For evaluation, we run all the methods on 100 testing instances for each dataset.

4.2 Main Results on Cross-Domain Generalization

We first evaluate the effectiveness of RoME trained on multiple domains on various test domains. In this experiment, RoME is trained on IS, IP, and SC. We train just one unified model on these three datasets, and then test it on all five domains, where WA and CA serve as unseen zero-shot testing datasets. The baseline methods are trained on each domain individually and evaluated on the corresponding testing datasets. Table 1 presents the average objective values and gap_{abs} achieved by each method using the Gurobi solver across all five combinatorial problem families. Corresponding results for SCIP are provided in Appendix D.2. For the IS dataset, we also report the average solver time in seconds, as all the methods can find the optimal solution within the time limit. The results in Table 1 show that RoME consistently outperforms all baselines across domains. (1) For IS, IP and

SC, RoME is able to achieve the best objective values and absolute primal gap than the baselines trained on the corresponding dataset. This implies that cross-domain training can benefit from the performance of each single domain. (2) Notably, even though not trained on WA and CA, RoME achieves the best performance on these unseen domains, demonstrating its zero-shot generalization capability.

We also report the curves that the average primal gap (defined as $\text{gap}_{\text{rel}} := |\text{OBJ} - \text{BKS}|/|\text{BKS}|$) changes as the solving process proceeds in Figure 3. A rapid decrease in the curves indicates superior solving performance and better convergence. The results in Figure 3 show that RoME exhibits a rapid decline in primal gap and ultimately achieves the lowest primal gap.

Additionally,, we conduct comprehensive ablation studies to investigate the critical components of RoME, evaluating the effects of key elements including DRO, the MoE architecture, embedding perturbation strategies, and expert diversity. Detailed analyses can be found in Appendix D.5.

4.3 Zero-Shot Generalization to MIPLIB Problems

To further evaluate the performance of RoME in the real-world complex dataset, we evaluate it on a curated subset of the MIPLIB benchmark [10]. The MIPLIB dataset is a challenging MILP dataset, where the instances are drawn from diverse industrial applications with heterogeneity modeling structures. In this experiment, the baselines are trained and evaluated on splits within MIPLIB. In contrast, RoME is trained on synthetic datasets as used in the main evaluation and directly applied to MIPLIB in a zero-shot fashion. This setting simulates a practical scenario where the solver is deployed in unseen domains without any retraining.

We select two groups of MIPLIB testing instances following the selection criterion in Appendix C.3. The first subset follows Liu et al. [32] and uses the IIS dataset selected from MILPLIB. IIS contains six instances for baselines and five testing instances, where we draw the solving curve in Figure 4. And another subset contains fifteen challenging MIPLIB instances in total, with Figure 5 showcasing solving curves for a representative subset of five selected instances. We find RoME consistently outperforms all the baselines across all the instances, highlighting its strong generalization ability on real-world complex instances. Average results of objective values and gap_{abs} across both two selected MIPLIB datasets are presented in Appendix D.1, with each instance result also provided in the same appendix.

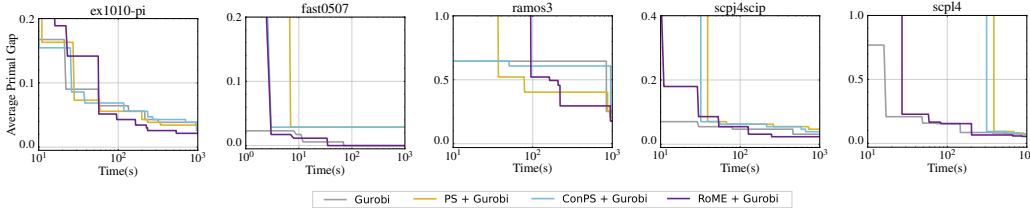


Figure 4: **The primal gap of different methods on five easier instances from IIS dataset as the solving process proceeds.** We use Gurobi for implementation and set the time limit to be 1,000 seconds.

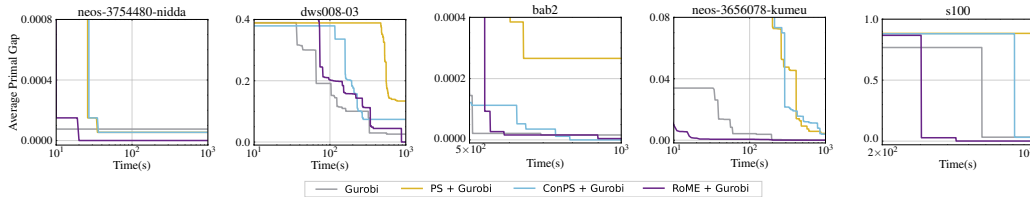


Figure 5: **The primal gap of different methods on five harder instances from MIPLIB as the solving process proceeds.** We use Gurobi for implementation and set the time limit to 1,000s.

4.4 Interpretability of RoME

Current works typically adopt GNN-based models, which are inherently treated as black-boxes. However, with the recent progress in large language models [50–52], an increasing number of interpretability techniques have emerged, particularly for MoE architectures, offering new opportunities

to better understand model behavior. Following this, to gain a deeper understanding of RoME, we visualise its expert routing and embedding behaviour. Using the model trained on IS, IP and SC following the main evaluation, we record which expert the gate selects when it solves instances from the five benchmark domains and from five selected challenging instances from MIPLIB, as summarised in Fig.6 (a). We then obtain the task vectors of these instance and plot their distribution through T-SNE, as shown in Fig.6 (b). These interpretability analyses reveal that RoME develops a clear and structured specialization among its experts. Each problem with unique constraint type consistently activates a distinct expert, demonstrating that the mixture model avoids collapse and that individual experts capture unique structural regularities across tasks. The alignment between task embeddings and expert activations shows that RoME learns coherent, task-aware representations: tasks with similar underlying formulations are embedded closely in the latent space and routed to overlapping expert subsets. Furthermore, mixed-constraint MIPLIB instances are positioned near and assigned to the domains they most resemble, explaining RoME’s strong generalization and zero-shot adaptability to previously unseen problem families.

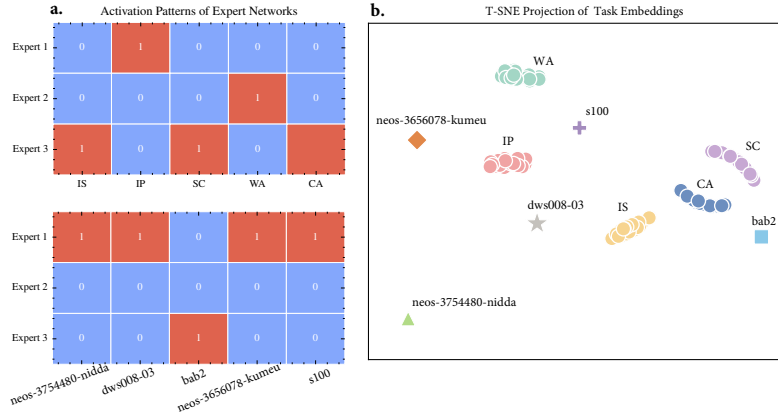


Figure 6: **Model interpretability.** (a) Expert network activation patterns across different MILP instances. Red indicates activated experts, while blue denotes inactive ones. (b) The distribution of instances in the task embedding space.

5 Conclusion

In this paper, we present RoME, a cross-domain MILP solver that pairs a mixture-of-experts architecture with a distributionally robust training objective. The mixture-of-experts architecture enhance representation ability across different domains, while distributionally robust optimization shields against distribution shifts, and together they boost generalization across MILP problem families. Experiments demonstrate substantial improvements over existing learning-based baselines and strong zero-shot performance on challenging MIPLIB instances.

Acknowledgments and Disclosure of Funding

The authors would like to thank all the anonymous reviewers for their valuable suggestions. This work was supported by the National Natural Science Foundation of China (NSFC, 72571284, 72421002, 62206303, 62273352), the Hunan Provincial Fund for Distinguished Young Scholars (2025JJ20073), the Science and Technology Innovation Program of Hunan Province (2023RC3009), the Foundation Fund Program of National University of Defense Technology (JS24-05), the Major Science and Technology Projects in Changsha, China (kq2301008), the Key Research and Development Program of Hunan Province (2025JK2143), and the Hunan Provincial Innovation Foundation for Postgraduate (XJQY2025044).

References

- [1] Reinhard Baller, Pirmin Fontaine, Stefan Minner, and Zhen Lai. Optimizing automotive inbound logistics: A mixed-integer linear programming approach. *Transportation Research Part E: Logistics and Transportation Review*, 163:102734, 2022. ISSN 1366-5545. doi: <https://doi.org/10.1016/j.tre.2022.102734>. URL <https://www.sciencedirect.com/science/article/pii/S1366554522001259>.
- [2] Mahdi Sharifzadeh, Marti Cortada Garcia, and Nilay Shah. Supply chain network design and operation: Systematic decision-making for centralized, distributed, and mobile biofuel production using mixed integer linear programming (milp) under uncertainty. *Biomass and Bioenergy*, 81: 401–414, 2015. ISSN 0961-9534. doi: <https://doi.org/10.1016/j.biombioe.2015.07.026>. URL <https://www.sciencedirect.com/science/article/pii/S0961953415300659>.
- [3] David M Ryan and Brian A Foster. An integer programming approach to scheduling. *Computer scheduling of public transport urban passenger vehicle and crew scheduling*, pages 269–280, 1981.
- [4] S. Belil, S. Kemmoé-Tchomté, and N. Tchernev. Milp-based approach to mid-term production planning of batch manufacturing environment producing bulk products. *IFAC-PapersOnLine*, 51(11):1689–1694, 2018. ISSN 2405-8963. doi: <https://doi.org/10.1016/j.ifacol.2018.08.213>. URL <https://www.sciencedirect.com/science/article/pii/S240589631831334X>. 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018.
- [5] Tianle Pu, Chao Chen, Li Zeng, Shixuan Liu, Rui Sun, and Changjun Fan. Solving combinatorial optimization problem over graph through qubo transformation and deep reinforcement learning. In *2024 IEEE International Conference on Data Mining (ICDM)*, pages 390–399. IEEE, 2024.
- [6] Tianle Pu, Changjun Fan, Mutian Shen, Yizhou Lu, Li Zeng, Zohar Nussinov, Chao Chen, and Zhong Liu. Exploratory combinatorial optimization problem solving via gauge transformation. In *2024 IEEE International Conference on Data Mining (ICDM)*, pages 821–826. IEEE, 2024.
- [7] Haopeng Duan, Kaiming Xiao, Lihua Liu, Haiwen Chen, and Hongbin Huang. Reinforce with bound-guided gradient estimator for the traveling salesman problem toward scale generalization. *Engineering Applications of Artificial Intelligence*, page 109819, 2025.
- [8] LLC Gurobi Optimization. Gurobi optimizer. URL <http://www.gurobi.com>, 2021.
- [9] Tobias Achterberg. Scip: solving constraint integer programs. *Mathematical Programming Computation*, 1:1–41, 2009.
- [10] Ambros Gleixner, Gregor Hendel, Gerald Gamrath, Tobias Achterberg, Michael Bastubbe, Timo Berthold, Philipp Christophel, Kati Jarck, Thorsten Koch, Jeff Linderoth, et al. Miplib 2017: data-driven compilation of the 6th mixed-integer programming library. *Mathematical Programming Computation*, 13(3):443–490, 2021.
- [11] Maxime Gasse, Didier Chételat, Nicola Ferroni, Laurent Charlin, and Andrea Lodi. Exact combinatorial optimization with graph convolutional neural networks. *Advances in neural information processing systems*, 32, 2019.
- [12] Zhihai Wang, Xijun Li, Jie Wang, Yufei Kuang, Mingxuan Yuan, Jia Zeng, Yongdong Zhang, and Feng Wu. Learning cut selection for mixed-integer linear programming via hierarchical sequence model. In *The Eleventh International Conference on Learning Representations*, 2023.
- [13] Yufei Kuang, Jie Wang, Haoyang Liu, Fangzhou Zhu, Xijun Li, Jia Zeng, HAO Jianye, Bin Li, and Feng Wu. Rethinking branching on exact combinatorial optimization solver: The first deep symbolic discovery framework. In *The Twelfth International Conference on Learning Representations*, 2024.
- [14] Hongyu Liu, Haoyang Liu, Yufei Kuang, Jie Wang, and Bin Li. Deep symbolic optimization for combinatorial optimization: Accelerating node selection by discovering potential heuristics. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 2067–2075, 2024.

- [15] Nicolas Sonnerat, Pengming Wang, Ira Ktena, Sergey Bartunov, and Vinod Nair. Learning a large neighborhood search algorithm for mixed integer programs. *ArXiv*, abs/2107.10201, 2021.
- [16] Taoan Huang, Aaron Ferber, Yuandong Tian, Bistra N. Dilkina, and Benoit Steiner. Searching large neighborhoods for integer linear programs with contrastive learning. In *International Conference on Machine Learning*, 2023.
- [17] Vinod Nair, Sergey Bartunov, Felix Gimeno, Ingrid Von Glehn, Pawel Lichocki, Ivan Lobov, Brendan O’Donoghue, Nicolas Sonnerat, Christian Tjandraatmadja, Pengming Wang, et al. Solving mixed integer programs using neural networks. *arXiv preprint arXiv:2012.13349*, 2020.
- [18] Qingyu Han, Linxin Yang, Qian Chen, Xiang Zhou, Dong Zhang, Akang Wang, Ruoyu Sun, and Xiaodong Luo. A gnn-guided predict-and-search framework for mixed-integer linear programming. In *The Eleventh International Conference on Learning Representations*, 2023.
- [19] Taoan Huang, Aaron M Ferber, Arman Zharmagambetov, Yuandong Tian, and Bistra Dilkina. Contrastive predict-and-search for mixed integer linear programs. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 19757–19771. PMLR, 21–27 Jul 2024. URL <https://proceedings.mlr.press/v235/huang24f.html>.
- [20] Elias B Khalil, Christopher Morris, and Andrea Lodi. Mip-gnn: A data-driven framework for guiding combinatorial solvers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10219–10227, 2022.
- [21] Zijie Geng, Xijun Li, Jie Wang, Xiao Li, Yongdong Zhang, and Feng Wu. A deep instance generative framework for milp solvers under limited data availability. In *Advances in Neural Information Processing Systems*, 2023.
- [22] Haoyang Liu, Yufei Kuang, Jie Wang, Xijun Li, Yongdong Zhang, and Feng Wu. Promoting generalization for exact solvers via adversarial instance augmentation, 2023. URL <https://arxiv.org/abs/2310.14161>.
- [23] Haoyang Liu, Jie Wang, Wanbo Zhang, Zijie Geng, Yufei Kuang, Xijun Li, Bin Li, Yongdong Zhang, and Feng Wu. MILP-studio: MILP instance generation via block structure decomposition. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL <https://openreview.net/forum?id=W433RI0VU4>.
- [24] Qingyu Qu, Xijun Li, Yunfan Zhou, Jia Zeng, Mingxuan Yuan, Jie Wang, Jinhu Lv, Kexin Liu, and Kun Mao. An improved reinforcement learning algorithm for learning to branch. *arXiv preprint arXiv:2201.06213*, 2022.
- [25] He He, Hal Daume III, and Jason M Eisner. Learning to search in branch and bound algorithms. *Advances in neural information processing systems*, 27, 2014.
- [26] Abdel Ghani Labassi, Didier Chetelat, and Andrea Lodi. Learning to compare nodes in branch and bound with graph neural networks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 32000–32010. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/cf5bb18807a3e9cfaaa51e667e18f807-Paper-Conference.pdf.
- [27] Sijia Zhang, Shuli Zeng, Shaoang Li, Feng Wu, and Xiangyang Li. Learning to select nodes in branch and bound with sufficient tree representation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=gyvYKLEm8t>.
- [28] Sirui Li, Wenbin Ouyang, Max B Paulus, and Cathy Wu. Learning to configure separators in branch-and-cut. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

- [29] Jialin Song, ravi lanka, Yisong Yue, and Bistra Dilkina. A general large neighborhood search framework for solving integer linear programs. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 20012–20023. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/e769e03a9d329b2e864b4bf4ff54ff39-Paper.pdf.
- [30] Yaixin Wu, Wen Song, Zhiguang Cao, and Jie Zhang. Learning large neighborhood search policy for integer programming. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 30075–30087. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/fc9e62695def29ccdb9eb3fed5b4c8c8-Paper.pdf.
- [31] Huigen Ye, Hua Xu, Hongyan Wang, Chengming Wang, and Yu Jiang. GNN&GBDT-guided fast optimizing framework for large-scale integer programming. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 39864–39878. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/ye23e.html>.
- [32] Haoyang Liu, Jie Wang, Zijie Geng, Xijun Li, Yuxuan Zong, Fangzhou Zhu, Jianye HAO, and Feng Wu. Apollo-MILP: An alternating prediction-correction neural solving framework for mixed-integer linear programming. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=mFY0tPDWK8>.
- [33] Zijie Geng, Jie Wang, Xijun Li, Fangzhou Zhu, Jianye HAO, Bin Li, and Feng Wu. Differentiable integer linear programming. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=FPfCUJTscn>.
- [34] Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42:275–293, 2014.
- [35] Rui Gao, Xi Chen, and Anton J Kleywegt. Wasserstein distributionally robust optimization and variation regularization. *Operations Research*, 72(3):1177–1191, 2024.
- [36] Daniel Kuhn, Peyman Mohajerin Esfahani, Viet Anh Nguyen, and Soroosh Shafieezadeh-Abadeh. Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations research & management science in the age of analytics*, pages 130–166. INFORMS, 2019.
- [37] Jiang Guo, Darsh J Shah, and Regina Barzilay. Multi-source domain adaptation with mixture of experts. *arXiv preprint arXiv:1809.02256*, 2018.
- [38] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.
- [39] Bo Li, Yifei Shen, Jingkan Yang, Yezhen Wang, Jiawei Ren, Tong Che, Jun Zhang, and Ziwei Liu. Sparse mixture-of-experts are domain generalizable learners. *arXiv preprint arXiv:2206.04046*, 2022.
- [40] Jianan Zhou, Zhiguang Cao, Yaixin Wu, Wen Song, Yining Ma, Jie Zhang, and Chi Xu. Mvmoe: Multi-task vehicle routing solver with mixture-of-experts, 2024. URL <https://arxiv.org/abs/2405.01029>.
- [41] Fei Liu, Xi Lin, Zhenkun Wang, Qingfu Zhang, Xialiang Tong, and Mingxuan Yuan. Multi-task learning for routing problem with cross-problem zero-shot generalization, 2024. URL <https://arxiv.org/abs/2402.16891>.
- [42] Yang Li, Jiayi Liu, Qitian Wu, Xiaohan Qin, and Junchi Yan. Toward learning generalized cross-problem solving strategies for combinatorial optimization, 2025.

- [43] Darko Drakulic, Sofia Michel, and Jean-Marc Andreoli. GOAL: A generalist combinatorial optimization agent learner. In *The Thirteenth International Conference on Learning Representations*, 2025.
- [44] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.
- [45] Jingge Wang, Yang Li, Liyan Xie, and Yao Xie. Class-conditioned domain generalization via wasserstein distributional robust optimization. *arXiv preprint arXiv:2109.03676*, 2021.
- [46] Rui Dai, Yonggang Zhang, Zhen Fang, Bo Han, and Xinmei Tian. Moderately distributional exploration for domain generalization. *arXiv preprint arXiv:2304.13976*, 2023.
- [47] Lijun Zhang, Peng Zhao, Zhen-Hua Zhuang, Tianbao Yang, and Zhi-Hua Zhou. Stochastic approximation approaches to group distributionally robust optimization. *Advances in Neural Information Processing Systems*, 36:52490–52522, 2023.
- [48] 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 *The Thirteenth International Conference on Learning Representations*, 2025.
- [49] Maxime Gasse, Simon Bowly, Quentin Cappart, Jonas Charfreitag, Laurent Charlin, Didier Chételat, Antonia Chmiela, Justin Dumouchelle, Ambros Gleixner, Aleksandr M. Kazachkov, Elias Khalil, Pawel Lichocki, Andrea Lodi, Miles Lubin, Chris J. Maddison, Morris Christopher, Dimitri J. Papageorgiou, Augustin Parjadis, Sebastian Pokutta, Antoine Prouvost, Lara Scavuzzo, Giulia Zarpellon, Linxin Yang, Sha Lai, Akang Wang, Xiaodong Luo, Xiang Zhou, Haohan Huang, Shengcheng Shao, Yuanming Zhu, Dong Zhang, Tao Quan, Zixuan Cao, Yang Xu, Zhewei Huang, Shuchang Zhou, Chen Binbin, He Minggui, Hao Hao, Zhang Zhiyu, An Zhiwu, and Mao Kun. The machine learning for combinatorial optimization competition (ml4co): Results and insights. In Douwe Kiela, Marco Ciccone, and Barbara Caputo, editors, *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*, volume 176 of *Proceedings of Machine Learning Research*, pages 220–231. PMLR, 06–14 Dec 2022. URL <https://proceedings.mlr.press/v176/gasse22a.html>.
- [50] Jincai Huang, Yongjun Xu, Qi Wang, Qi Cheems Wang, Xingxing Liang, Fei Wang, Zhao Zhang, Wei Wei, Boxuan Zhang, Libo Huang, et al. Foundation models and intelligent decision-making: Progress, challenges, and perspectives. *The Innovation*, 2025.
- [51] Rui Wang, Shixuan Liu, Changjun Fan, Guozheng Li, Jincai Huang, Zhong Liu, and Gang Zhou. Towards bridging the gap between machine intelligence and machine wisdom: dilemmas and conjectures. *The Innovation*, page 100834, 2025.
- [52] Liu Haoyang, Wang Jie, Cai Yuyang, Han Xiongwei, Kuang Yufei, and Hao Jianye. Optitree: Hierarchical thoughts generation with tree search for LLM optimization modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.
- [53] Changjun Fan, Mutian Shen, Zohar Nussinov, Zhong Liu, Yizhou Sun, and Yang-Yu Liu. Searching for spin glass ground states through deep reinforcement learning. *Nature communications*, 14(1):725, 2023.
- [54] Changjun Fan, Li Zeng, Yuhui Ding, Muhao Chen, Yizhou Sun, and Zhong Liu. Learning to identify high betweenness centrality nodes from scratch: A novel graph neural network approach. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 559–568, 2019.
- [55] Changjun Fan, Li Zeng, Yizhou Sun, and Yang-Yu Liu. Finding key players in complex networks through deep reinforcement learning. *Nature machine intelligence*, 2(6):317–324, 2020.
- [56] Shixuan Liu, Changjun Fan, Kewei Cheng, Yunfei Wang, Peng Cui, Yizhou Sun, and Zhong Liu. Inductive meta-path learning for schema-complex heterogeneous information networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

- [57] Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4732–4740, 2021.
- [58] Hong Wang, Jie Wang, Minghao Ma, Haoran Shao, and Haoyang Liu. Symmap: Improving computational efficiency in linear solvers through symbolic preconditioning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=0upeovfx0L>.
- [59] Hong Wang, Haoyang Liu, Jian Luo, Jie Wang, et al. Accelerating pde data generation via differential operator action in solution space. In *Forty-first International Conference on Machine Learning*.
- [60] Shixuan Liu, Yue He, Haotian Wang, Wenjing Yang, Yunfei Wang, Peng Cui, and Zhong Liu. Environment inference for learning generalizable dynamical systems. In *Advances in Neural Information Processing Systems*, volume 39, 2025.
- [61] Chang Liu, Yunfan Ye, Fan Zhang, Qingyang Zhou, Yuchuan Luo, and Zhiping Cai. Humansam: Classifying human-centric forgery videos in human spatial, appearance, and motion anomaly. *arXiv preprint arXiv:2507.19924*, 2025.
- [62] Li Liu, Songyang Lao, Paul W. Fieguth, Yulan Guo, Xiaogang Wang, and Matti Pietikäinen. Median robust extended local binary pattern for texture classification. *IEEE Transactions on Image Processing*, 25(3):1368–1381, 2016.
- [63] Li Liu, Shuzhou Sun, Shuaifeng Zhi, Fan Shi, Zhen Liu, Janne Heikkilä, and Yongxiang Liu. A causal adjustment module for debiasing scene graph generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(5):4024–4043, 2025.
- [64] Weijie Li, Wei Yang, Yuenan Hou, Li Liu, Yongxiang Liu, and Xiang Li. Saratr-x: Toward building a foundation model for sar target recognition. *IEEE Transactions on Image Processing*, 34(1):869–884, 2025. doi: 10.1109/TIP.2025.3531988.
- [65] Bingxu Zhang, Changjun Fan, Shixuan Liu, Kuihua Huang, Xiang Zhao, Jincai Huang, and Zhong Liu. The expressive power of graph neural networks: A survey. *IEEE Transactions on Knowledge & Data Engineering*, (01):1–20, 2024.
- [66] Kevin Leyton-Brown, Mark Pearson, and Yoav Shoham. Towards a universal test suite for combinatorial auction algorithms. In *Proceedings of the 2nd ACM Conference on Electronic Commerce*, pages 66–76, 2000.
- [67] Egon Balas and Andrew Ho. *Set covering algorithms using cutting planes, heuristics, and subgradient optimization: a computational study*. Springer, 1980.
- [68] Maxime Gasse, Simon Bowly, Quentin Cappart, Jonas Charfreitag, Laurent Charlin, Didier Chételat, Antonia Chmiela, Justin Dumouchelle, Ambros Gleixner, Aleksandr M Kazachkov, et al. The machine learning for combinatorial optimization competition (ml4co): Results and insights. In *NeurIPS 2021 competitions and demonstrations track*, pages 220–231. PMLR, 2022.

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 main claims can reflect the paper's contribution and scope.

Guidelines:

- The answer NA 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 NA 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: Please see Appendix E.1.

Guidelines:

- The answer NA 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: [\[Yes\]](#)

Justification: Section 3.3 and 3.4 introduce our DRO method in detail.

Guidelines:

- The answer NA 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 the implementation details in Appendix A.

Guidelines:

- The answer NA 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: [No]

Justification: Code is available at <https://github.com/happypu326/RoME>.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.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://nips.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, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We specify the training and test details in Section 4.1, the hyperparameters in Appendix B and benchmark details in Appendix C.

Guidelines:

- The answer NA 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: [Yes]

Justification: We analyze the experiment results in Section 4.

Guidelines:

- The answer NA 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 report the machine we used in Appendix A.3.

Guidelines:

- The answer NA 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: We obey the NeurIPS Code of Ethics in the research.

Guidelines:

- The answer NA 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: [Yes]

Justification: We discuss in Appendix E.3.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks.

Guidelines:

- The answer NA 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: We cite the dataset and code we used in Section 4.1 and Appendix A.3.

Guidelines:

- The answer NA 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: [NA]

Justification: The paper does not release new assets.

Guidelines:

- The answer NA 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: [NA]

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

Guidelines:

- The answer NA 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: [NA]

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

Guidelines:

- The answer NA 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 rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

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

Guidelines:

- The answer NA 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 (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

A Implementation Details

A.1 Bipartite Graph Representation

Following previous works [18, 32], we represent each MILP instance as a weighted bipartite graph $\mathcal{G} = (\mathcal{V} \cup \mathcal{W}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{W}$ denote the sets of variables and constraints, respectively. The features attached to the bipartite graph are similar to the previous works [18, 32]. The detailed description of these features can be found in Table 2.

Table 2: Description of the variable, Constraint and edge features used in our bipartite graph representation.

Index	Variable Feature Name	Description
0	Objective	Normalized objective coefficient
1	Variable coefficient	Average variable coefficient in all constraints
2	Variable degree	Degree of the variable node in the bipartite graph representation
3	Maximum variable coefficient	Maximum variable coefficient in all constraints
4	Minimum variable coefficient	Minimum variable coefficient in all constraints
5	Variable type	Whether the variable is an integer variable or not)
Index	Constraint Feature Name	Description
0	Constraint coefficient	Average of all coefficients in the constraint
1	Constraint degree	Degree of constraint nodes
2	Bias	Normalized right-hand-side of the constraint
3	Sense	The sense of the constraint
Index	Constraint Feature Name	Description
0	Coefficient	Constraint coefficient

A.2 Implementation Details of the GNN Encoder

In this work, we use a graph neural network (GNN) parameterized by θ to extract diverse structural representations for MILP instances. For a given instance $\mathcal{I}$, we initialize features for variables $v_i \in \mathbb{R}^6$, constraints $w_j \in \mathbb{R}^4$, and edges $e_{ij} \in \mathbb{R}^1$, then embed them through the MLP layers:

$$h_{v_i}^{(0)} = \text{MLP}_{\theta}(v_i), \quad h_{w_i}^{(0)} = \text{MLP}_{\theta}(w_i), \quad h_{e_{ij}}^{(0)} = \text{MLP}_{\theta}(e_{ij}). \quad (18)$$

Subsequently, we apply two graph convolution layers, each consisting of two interleaved half-convolution operations [11]:

$$\begin{aligned} h_{w_i}^{(k+1)} &\leftarrow \text{MLP}_{\theta} \left(h_{w_i}^{(k)}, \sum_{j:e_{ij} \in \mathcal{E}} \text{MLP}_{\phi}(h_{w_i}^{(k)}, h_{e_{ij}}, h_{v_j}^{(k)}) \right) \\ h_{v_i}^{(k+1)} &\leftarrow \text{MLP}_{\phi} \left(h_{v_i}^{(k)}, \sum_{i:e_{ij} \in \mathcal{E}} \text{MLP}_{\theta}(h_{w_i}^{(k+1)}, h_{e_{ij}}, h_{v_j}^{(k)}) \right) \end{aligned} \quad (19)$$

Unlike the previous works that employ extra MLPs for logit prediction, our GNN encoder directly propagates the refined node embeddings to the following modules.

A.3 Implementation Details of the Baselines

Optimization has broad applications in the real world, including statistical physics [53], complex networks [54, 55], knowledge graph [56, 57], and partial differential equations [58–60]. Recent breakthroughs in computer vision [61–64], graph models [65], and reinforcement learning have

propelled the adoption of machine learning for faster combinatorial optimization solving. In our work, we select two state-of-the-art methods, PS [18] and ConPS [19], as baseline approaches, each exemplifying a prominent direction in recent research. For PS, we utilize the official implementation available at <https://github.com/sribdcn/Predict-and-Search>, and we incorporate the same graph neural network (GNN) encoder from PS into our model architecture. Since ConPS does not have publicly available source code, we meticulously reimplemented its framework based on the original descriptions and performed extensive hyperparameter tuning to replicate its reported performance. All experiments are conducted on a single machine equipped with NVIDIA GeForce RTX 3090 GPUs and AMD EPYC 7402 24-core CPUs running at 2.80GHz. We use Gurobi version 11.0.3 and SCIP version 8.1.0 in all experiments.

For each baseline, we collect 300 instances to serve as training and validation data. Each instance is solved using Gurobi with a single thread for 3600 seconds, during which we record the best 50 solutions. We split the data into 80% for training and 20% for validation, and designate an additional 100 instances as the test set. During training, we set the initial learning rate to 0.0005 and train the model for 10,000 epochs, employing an early stopping mechanism to prevent overfitting. In the testing phase, the parameters (k_0, k_1, Δ) determine which variables are fixed to 0 or 1 and define the scope of the subsequent search process. These parameters significantly influence the performance of the methods. These parameters used in this work are detailed in Table 3.

Table 3: The partial solution size parameter (k_0, k_1) and neighborhood parameter Δ .

Benchmark	IS	CA	SC	IP	WA
PS+Gurobi	(300, 300, 20)	(600,0,1)	(2000,0,100)	(400,5,10)	(0,500,10)
ConPS+Gurobi	(1200, 600, 10)	(900,0,50)	(1000,0,200)	(400,5,3)	(0,500,10)
RoME+Gurobi	(250, 200, 15)	(350, 0, 55)	(1000, 0, 200)	(60, 35, 55)	(20, 200, 100)
PS+SCIP	(300,300, 15)	(400,0,10)	(2000,0,100)	(400,5,1)	(0,600,5)
ConPS+SCIP	(1200, 600, 10)	(900,0,50)	(1000,0,200)	(400,5,3)	(0,400,50)
RoME+SCIP	(250, 200, 15)	(350, 0, 55)	(1000, 0, 200)	(60, 35, 55)	(20, 200, 100)

B Hyperparameters

The key training parameters are summarized in Table 4.

Table 4: Hyperparameters used in our experiments.

Name	Value	Description
embed_size	128	The embedding size of the GNN encoder.
num_experts	3	Number of experts.
num_heads	3	Number of heads.
batch_size	4	Number of MILP instances in each training batch.
num_epochs	2000	Number of max running epochs.
lr	0.0005	Learning rate for training.
perturbation_ratio	0.1	Perturbation rate on the task embedding space.
expert_diversity_ratio	0.2	Expert diversity ratio for training.

C Details on the Benchmarks

C.1 Benchmarks in Main Evaluation

The CA, SC, and IS benchmark instances are generated following the procedure outlined in [11]. Specifically, the CA instances are created using the algorithm described in [66], while the SC instances are produced based on the method presented in [67]. The IP and WA instances are sourced from the NeurIPS ML4CO 2021 competition [68]. Table 5 provides detailed statistical information for all the instances.

Table 5: Statistical information of the benchmarks we used in this paper.

	CA	SC	IP	WA	IS
Constraint Number	2593	3000	195	64306	5943
Variable Number	1500	5000	1083	61000	1500
Number of Binary Variables	1500	5000	1050	1000	1500
Number of Continuous Variables	0	0	33	60000	0
Number of Integer Variables	0	0	0	0	0

C.2 Benchmarks used for Generalization

To evaluate the generalization capabilities of our methods, we construct larger CA and SC instances using the data generation code from Gasse et al. [11]. The newly generated CA instances contain approximately 2,596 constraints and 4,000 variables on average, while the SC instances comprise 6,000 constraints and 10,000 variables. These problem sizes are substantially larger than those encountered during training, providing a more rigorous evaluation of the models’ performance on unseen, large-scale instances.

C.3 Subset of MIPLIB

Table 6: Statistical information of the instances in the constructed MIPLIB dataset.

Instance Name	Constraint Number	Variable Number	Nonzero Coefficient Number
ex1010-pi	1468	25200	102114
fast0507	507	63009	409349
ramos3	2187	2187	32805
scpj4scip	1000	99947	999893
scpk4	2000	100000	1000000
scpl4	2000	200000	2000000
dws008-03	16344	32280	165168
dws008-01	6064	11096	56400
neos2	1103	2101	7326
bab2	17245	147912	2027726
bab5	4964	21600	155520
bab6	29904	114240	1283181
neos-3555904-turama	146493	37461	793605
neos-3656078-kumeu	17656	14870	59292
supportcase17	2108	1381	5253
fastxgemm-n2r7s4t1	6972	904	22584
neos-3754480-nidda	203	253	1488
bg512142	1307	792	3953
tr12-30	750	1080	2508
s100	14733	364417	1777917
s55	9892	78141	317902

To evaluate the solvers’ performance on challenging real-world instances, we construct two groups of MIPLIB [10] instances. The selection is based on instance similarity, measured using 100 human-designed features as described in [10]. Group 1 follows Liu et al [32], utilizing the IIS dataset from MIPLIB. This dataset comprises 11 instances, divided into a training set of six instances—glass-sc, iis-glass-cov, 5375, 214, 56133, iis-hc-cov, seymour, and v150d30-2hopcds—and a test set of five instances: ex1010-pi, fast0507, ramos3, scpj4scip, and scpk4. We note that baseline methods are trained on this split, whereas our method is evaluated in a zero-shot setting. To further evaluate generalization, we construct Group 2 by selecting five challenging instances from MIPLIB’s reported hard instances, dws008-03, bab2, neos-3656078-kumeu, neos-3754480-nidda, and s100, and then

identifying another 10 similar instances based on feature similarity to form the new group. Detailed information on the MIPLIB dataset is provided in Table 6.

D More Experiment Results

D.1 Results on MIPLIB Dataset

To further evaluate the generalization ability of RoME on MIPLIB, we conduct two group experiments. For the IIS subset, the baselines are trained on the designated training instances and then evaluated, whereas RoME is evaluated in a zero-shot setting. And for the remaining instances, every method is run in zero-shot mode. Since the instance sizes vary on different instances, we fix variables by a proportion, using $(k_0, k_1, \Delta) = (0.7, 0, 1000)$. We report two selected MIPLIB dataset from Section 4.3, each containing five selected instances, with the average performance across both datasets for each method presented in Table 7. The detailed MIPLIB results are presented in Table 8.

Table 7: The results in the IIS and the more complicated MIPLIB Instance. We build the ML approaches on Gurobi and set the solving time limit to 3,600s.

	IIS (BKS 196.00)		Hard Instance (BKS -58988.70)	
	Obj ↓	gap _{abs} ↓	Obj ↓	gap _{abs} ↓
Gurobi	211.00	15.00	-58663.30	325.40
PS+Gurobi	210.60	14.60	-57277.70	1710.99
ConPS+Gurobi	210.60	14.20	-58050.10	938.58
RoME+Gurobi	206.80	10.80	-58988.70	0.00

Table 8: The best objectives found by the approaches on each test instance in MIPLIB. *BKS* represents the best objectives from the website of MIPLIB <https://miplib.zib.de/index.html>.

	BKS	Gurobi	PS+Gurobi	ConPS+Gurobi	RoME+Gurobi
ex1010-pi	233.00	239.00	241.00	239.00	237.00
fast0507	174.00	174.00	179.00	179.00	174.00
ramos3	186.00	233.00	225.00	225.00	224.00
scpj4scip	128.00	132.00	133.00	133.00	131.00
scpl4	259.00	277.00	275.00	275.00	273.00
dws008-03	62831.76	64452.67	71234.06	67473.85	62831.75
dws008-01	37412.60	37412.60	39043.26	38817.50	37415.68
neos2	454.86	454.86	454.86	454.86	454.86
bab2	-357544.31	-357538.58	-357449.20	-357544.31	-357542.78
bab5	-106411.84	-106411.84	-106411.84	-106411.84	-106411.84
bab6	-284248.23	-284248.23	-284224.56	-284224.56	-284224.56
neos-3555904-turama	-34.7	-34.7	-34.7	-34.7	-34.7
neos-3656078-kumeu	-13172.2	-13171.3	-13114.0	-13120.6	-13172.2
supportcase17	1330.00	1330.00	1330.00	1330.00	1330.00
fastxgemm-n2r7s4t1	42.00	42.00	42.00	42.00	42.00
neos-3754480-nidda	12939.80	12940.78	12940.50	12940.50	12939.80
bg512142	184202.75	184202.75	190193.00	190193.00	190018.00
tr12-30	130596.00	130596.00	130596.00	130596.00	130596.00
s100	-0.1697	-0.1642	-0.0198	-0.1643	-0.1696
s55	-22.15	-22.15	-22.15	-22.15	-22.15

D.2 SCIP results in Main Evaluation

Consistent with the experimental setup in Section 4.2, we reported the results of different methods on the SCIP solver as shown in Table 9.

Table 9: Performance comparison across various MILP domains using SCIP solver, under a 1,000s time limit. We also report the improvement of our method over the traditional solvers in terms of gap_{abs} .

	IS (BKS 685.00)		IP (BKS 11.16)		SC (BKS 124.64)		WA (BKS 703.05)		CA (BKS 97524.37)	
	Obj $\uparrow$	Time $\downarrow$	Obj $\downarrow$	$\text{gap}_{\text{abs}} \downarrow$	Obj $\downarrow$	$\text{gap}_{\text{abs}} \downarrow$	Obj $\downarrow$	$\text{gap}_{\text{abs}} \downarrow$	Obj $\uparrow$	$\text{gap}_{\text{abs}} \downarrow$
SCIP	685.00	254.63	16.55	5.39	126.43	1.38	705.77	2.72	96423.90	1100.47
PS+SCIP	685.00	129.35	16.18	5.02	126.65	1.60	705.21	2.16	96426.46	1097.91
ConPS+SCIP	685.00	78.53	16.16	5.00	126.40	1.35	705.21	2.16	96428.83	1095.54
RoME+SCIP	685.00	13.55	16.08	4.92	126.27	1.22	705.08	2.03	96439.82	1084.55
Improvement		18.7 $\times$		8.7%		11.6%		25.3%		1.4%

D.3 Fine-grained Instance-level Results

To provide a fine-grained, instance-level view of RoME’s performance, we quantify its performance drop relative to Gurobi. Following standard practice in multi-task learning, we introduce a *win count* metric, the number of test instances, out of 100, on which each method achieves the best objective value. We first compare Gurobi, PS, and RoME on the CA and SC benchmarks. As shown in Table 10, RoME consistently dominates both benchmarks.

Table 10: Total win counts. # denotes the number of instances on which each method achieves the best objective.

	CA (wins)	SC (wins)
Gurobi	26/100	5/100
PS+Gurobi	19/100	10/100
RoME+Gurobi	55/100	85/100

We further separate the instances on which each learning-enhanced solver *wins* or *loses* against Gurobi and report the mean absolute objective gap (Gap_{abs}) in Table 11. The results show that RoME not only wins on a much larger portion of instances but also yields greater improvements when it wins and smaller degradations when it loses, confirming its stability and overall superiority.

Table 11: Average win/lose gap against Gurobi.

	CA	SC
PS+Gurobi win	178.94 (43/100)	0.18 (68/100)
PS+Gurobi lose	207.17 (57/100)	0.41 (32/100)
RoME+Gurobi win	295.28 (65/100)	0.37 (91/100)
RoME+Gurobi lose	114.93 (35/100)	0.10 (9/100)

D.4 Generalization Results

To evaluate the generalization capabilities of our approach, we evaluate its performance on larger instances of the CA and SC problems. These instances, detailed in Appendix C.2, are significantly larger than those used during training. We utilize the model trained on the dataset described in Section 4.1 to perform zero-shot evaluations on these larger instances. The results, presented in Table 12, indicate that our method, RoME, consistently outperforms baseline methods on these larger instances, demonstrating its strong generalization ability.

D.5 Ablation Studies

The Effect of DRO and MoE To clarify the contribution of each component in RoME, we compare the full model with three ablated variants: (i) an MoE-only version trained without the DRO, (ii) a DRO-only version that replaces the MoE with a GCN proposed in the PS [18], and (iii) a single-head variant that removes expert diversity within the MoE architecture. As reported in Table 13, omitting

Table 12: We evaluate the generalization performance on 100 larger instances with a 1,000s time limit.

	SC (BKS 101.45)		CA (BKS 115787.97)	
	Obj ↓	gap _{abs} ↓	Obj ↑	gap _{abs} ↓
Gurobi	102.29	0.84	114960.25	827.72
PS+Gurobi	102.27	0.82	115228.20	559.77
ConPS+Gurobi	102.18	0.73	115343.23	444.74
RoME+Gurobi	102.03	0.58	115787.97	0.00

either DRO or MoE substantially degrades performance, confirming that these two modules play indispensable roles in RoME’s cross-domain generalisation.

Table 13: Impact of DRO and MoE components under a 1,000s time limit. We report the average best objective values and absolute primal gap.

	IP (BKS 11.16)		SC (BKS 124.64)		WA (BKS 703.05)		CA (BKS 97524.37)	
	Obj ↓	gap _{abs} ↓	Obj ↓	gap _{abs} ↓	Obj ↓	gap _{abs} ↓	Obj ↑	gap _{abs} ↓
Gurobi	11.43	0.27	125.21	0.57	703.47	0.42	97308.93	215.44
MoE-only+Gurobi	11.27	0.11	124.65	0.01	708.33	5.18	95282.51	2241.86
DRO-only+Gurobi	11.27	0.11	124.64	0.00	704.13	1.08	97194.98	329.39
Single-head+Gurobi	11.18	0.02	124.83	0.19	703.14	1.09	92248.18	5276.19
RoME+Gurobi	11.22	0.06	124.69	0.05	703.11	0.06	97489.28	35.09

The Effect of Task Embedding Perturbation Furthermore, we evaluate how perturbation influences cross-domain generalization. In RoME, the perturbation is applied in the task-embedding space. We compare this design with three variants: one without any perturbation, one that perturbs the variable-embedding space, and one that perturbs the raw feature space. Table 14 shows that perturbing task embeddings yields the most robust predictions. This aligns with our expectation: the task-embedding space offers a higher-level representation of the entire instance, whereas perturbations in the variable-embedding or feature spaces can distort the underlying problem structure, leading to performance degradation, particularly when transferring across domains.

Table 14: Comparison of different perturbation locations under a 1,000s time limit. We report the average best objective values and absolute primal gap.

	IP (BKS 11.16)		SC (BKS 124.64)		WA (BKS 703.05)		CA (BKS 97524.37)	
	Obj ↓	gap _{abs} ↓	Obj ↓	gap _{abs} ↓	Obj ↓	gap _{abs} ↓	Obj ↑	gap _{abs} ↓
Gurobi	11.43	0.27	125.21	0.57	703.47	0.42	97308.93	215.44
RoME without Perturb Embedding	11.61	0.45	125.27	0.63	703.20	0.15	94898.67	2625.7
Perturb on Variable Embedding	11.16	0.00	124.84	0.20	703.24	0.19	94302.71	3221.66
Perturb on Variable Features	11.23	0.01	124.74	0.10	703.10	0.05	91472.19	6052.18
RoME+Gurobi	11.22	0.06	124.69	0.05	703.11	0.06	97489.28	35.09

We also explore different perturbation magnitudes. As shown in Table 15, a perturbation ratio of 0.1–0.15 yields the best trade-off.

The Effect of Expert Diversity Finally, we study the diversity of experts during training. Table 16 shows that appropriate augmentation of expert diversity improves model performance.

Table 15: Effect of perturbation ratio on CA and SC. The ML approaches are implemented using Gurobi, with a time limit set to 1,000s. ‘↑’ indicates that higher is better, and ‘↓’ indicates that lower is better. We mark the **best values** in bold.

Perturbation Ratio	SC (BKS 124.64)		CA (BKS 97524.37)	
	Obj ↓	gap _{abs} ↓	Obj ↑	gap _{abs} ↓
0.05	124.70	0.06	91362.30	6162.07
0.10	124.69	0.05	97489.28	35.09
0.15	124.66	0.02	97374.32	150.05
0.20	124.70	0.06	93362.68	4161.69

Table 16: Effect of expert diversity on CA and SC. The ML approaches are implemented using Gurobi, with a time limit set to 1,000s. ‘↑’ indicates that higher is better, and ‘↓’ indicates that lower is better. We mark the **best values** in bold.

Expert Diversity Ratio	SC (BKS 124.64)		CA (BKS 97524.37)	
	Obj ↓	gap _{abs} ↓	Obj ↑	gap _{abs} ↓
0.1	125.27	0.63	94898.67	2625.70
0.2	124.69	0.05	97489.28	35.09

E Discussions

E.1 Limitations

Higher training cost. Although cross-domain solvers demonstrate strong performance, their training requires substantial diverse data and incurs relatively high computational costs. Limited by GPU memory capacity, we selected 300 samples per cross-domain dataset. Our observations indicate that while model performance improves with larger training sets, this comes at the cost of significantly increased computational overhead.

Waiting for more sophisticated designs. Cross-domain MILP solving remains in its early stage and currently lacks sophisticated architectural designs. For instance, the field would benefit from more efficient sparse MoE architectures specifically optimized for large-scale MILP applications.

E.2 Future works

Better optimization algorithms. In this work, we employ DRO to balance training across different domains, while simulating cross-domain distribution shifts through perturbations in the task embedding space. We note that more sophisticated mathematical techniques could further enhance model robustness by explicitly incorporating uncertainty. Future research directions include developing advanced optimization algorithms to strengthen robustness and improve cross-domain generalization performance.

Better model architecture. Future research directions may explore more sophisticated MoE architectures to enhance the current framework, such as incorporating sparse MoE variants or attention mechanisms. Additionally, replacing the current GNN encoder with more expressive graph neural network models could further improve representation learning capabilities.

E.3 Broader Impacts

This paper presents RoME, a domain-robust Mixture-of-Experts framework that predicts high-quality initial solutions for diverse MILP problems without retraining. By plugging RoME into off-the-shelf solvers, practitioners in logistics, network design, and manufacturing can accelerate solution times and lower computational costs. Because RoME learns structural patterns rather than proprietary coefficients, it can be safely deployed on sensitive industrial instances while easing data-collection burdens and expanding access to learning-based optimization where labeled solutions are scarce.