# MIRAGE: Duality-Inspired MILP Augmentation for Representation Learning

**Ziao Guo**[1†], **Shiyue Wang**[2,3†], **Jiayuan Yang**[1], **Junchi Yan**[1,3 *]
[1]School of Computer Science & School of Artificial Intelligence, Shanghai Jiao Tong University
[2]School of Mathematical Sciences, Key Laboratory of MEA (Ministry of Education), Shanghai Key Laboratory of PMMP, and Nantong Institute for Applied Mathematics and Artificial Intelligence, East China Normal University
[3]Shanghai Innovation Institute
{ziao.guo, jessica123, yanjunchi}@sjtu.edu.cn wangshiyue@stu.ecnu.edu.cn

## Abstract

Learning high-quality representations of mixed-integer linear programming (MILP) is crucial for advancing machine learning (ML)-based methods, yet remains challenging due to the discrete nature of MILPs and their extreme sensitivity to small perturbations. Although contrastive learning has proven highly effective for representation learning in domains such as computer vision, designing effective data augmentations for MILPs is particularly difficult. To address this challenge, we propose MIRAGE, a principled data augmentation framework grounded in the invariance of the piecewise-affine dual price function induced by the branch-and-bound (B&B) algorithm. Viewing this dual function as a recorded, solver-level surrogate of the MILP's lower-bound structure, we derive two complementary augmentation certificates that act as its right-hand and objective-side projections. Both certificates are provably invariant under the same dual price function on the recorded B&B tree, and therefore preserve either the structural similarity of the search tree or the optimality of the original optimal solution. Empirically, we integrate MIRAGE with ML models for two significant downstream tasks—learning to branch and predict-and-search—and observe consistent performance improvements across multiple benchmarks. By grounding MILP data augmentation in a single, solver-induced duality quantity, our work provides a principled and effective approach to MILP representation learning.

## 1 Introduction

As one of the most fundamental paradigms for combinatorial optimization (CO), mixed-integer linear programming (MILP) is widely applied to real-world scenarios in scheduling [11], routing [7], and resource allocation [30], among others. Despite its versatility, solving large-scale MILP instances remains a significant challenge due to the NP-hardness of the problem and the concomitant exponential expansion of the search space. A promising direction to mitigate these computational barriers is the incorporation of machine learning (ML) techniques [5]. These approaches seek to harness common structures and patterns across MILP instances from a similar distribution, thereby guiding the solver to improve either computational efficiency or the quality of the obtained solutions.

*Correspondence author. † denotes equal contribution. This work was partly supported by National Cyber Security-National Science and Technology Major Project (No. 2026ZD1500700), NSFC (92370201, 12331014), and Shanghai Innovation Institute and the Shanghai Institute for Mathematics and Interdisciplinary Sciences (SIMIS) for the project (SIMIS-ID-26-AMS-001).

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

Owing to the discrete nature of MILPs, their solutions are often highly sensitive to perturbations. Even minor changes to the objective function or constraints can drastically alter a problem's computational complexity or, in many cases, lead to infeasibility. Furthermore, the ML4CO field prioritizes lightweight model architectures—often comprising just a few layers of graph neural networks (GNNs) [14, 21]—to meet stringent requirements for efficiency and low latency. Consequently, these models possess a parameter count significantly smaller than those typical in computer vision (CV) or natural language processing (NLP). This limited capacity can hinder the model's ability to capture subtle, yet critical, problem characteristics within MILP instances, which in turn constrains its performance on downstream MILP solving tasks.

An effective approach to mitigate this issue involves data augmentation combined with contrastive learning (CL), which has been shown to be effective in representation learning [25]. A fundamental challenge, however, arises in generating augmented MILP instances to serve as positive samples. This stems from the difficulty of designing perturbations that introduce meaningful variation without fundamentally altering the instance's core solution landscape. Consequently, the key is to identify a class of augmentation strategies to which the MILP's solution or solving strategy is robust, thereby navigating the tension between creating meaningful variations and preserving core problem properties. We identify that **duality** provides a rigorous basis for designing such perturbations.

The theory of duality in linear programming (LP) is well-established, having substantially contributed to both theoretical understanding and practical applications through efficient bound computation and algorithm design [3]. While many theoretical concepts from LP duality can be formally extended to MILP, their practical utility has been limited. A central challenge is the difficulty in formulating a standard dual for MILP that retains the desirable properties of its LP counterpart. Consequently, existing MILP dual formulations typically present a trade-off, being either insufficiently strong for effective bounding or computationally intractable. Despite this, MILP duality remains valuable for post-hoc sensitivity analysis, enabling the assessment of how changes to input data influence the optimal solution or its value [20].

In this paper, we leverage the piecewise-affine dual price function $F_{\mathrm{BB}}$ induced by the branch-and-bound (B&B) algorithm, which provides a strong dual for MILPs and is readily recorded during a routine run of a solver such as SCIP [1]. Our central observation is that $F_{\mathrm{BB}}$ is an *affine probe* of instance perturbations: right-hand-side shifts $\Delta b$ and objective shifts $\Delta c$ enter $F_{\mathrm{BB}}$ through the recorded duals $\lambda^t$ and $(\mu^t, \bar{\mu}^t)$ (Sec. 4.1). This motivates a unifying principle—perturbations that leave $F_{\mathrm{BB}}$ invariant on the recorded B&B tree preserve either the search-tree geometry or the nominal optimality. Building on this principle, we propose **MIRAGE**, a duality-inspired contrastive representation learning framework for MILPs, and derive two complementary certificates as the constraint- and objective-side projections of this invariance: an *inactive-basic RHS certificate* and a *box-normal objective certificate*, yielding positive samples that preserve the B&B tree or the original optimal solution, respectively. Matched random-sign perturbations, by contrast, provably incur non-trivial drift in $F_{\mathrm{BB}}$ (Appendix A.5) and serve as principled negatives. This common dual-invariance lens enables our model to isolate solution-critical instance features and learn high-quality representations for downstream solving. **Our core contributions are as follows:**

1) We identify invariance of the B&B dual price function $F_{\mathrm{BB}}$ as a unifying design principle for MILP data augmentation, and cast the problem of generating valid positive samples as that of characterizing perturbations along which $F_{\mathrm{BB}}$ is preserved on the recorded B&B tree.

2) Guided by this principle, we derive two complementary augmentation certificates: an inactive-basic RHS certificate and a box-normal objective certificate, as computable, instance-specific projections of the same dual invariance. These certificates convert recorded B&B basis/slack information or a nominal optimizer into executable positive regions for MILP contrastive learning. We prove that perturbations within these regions preserve either the B&B search-tree structure or the optimality of the nominal solution, and provide controlled evidence that certificate-constrained positives outperform support-cardinality- and norm-matched non-certified positives in the evaluated settings.

3) Building on these certificates, we propose MIRAGE, a duality-inspired contrastive representation learning framework for MILPs. Integrating MIRAGE with existing ML methods for learning-to-branch and predict-and-search yields consistent performance gains across multiple benchmarks.

## 2 Related Work

**ML-Enhanced MILP Solving.** ML has demonstrated significant potential for accelerating MILP. A substantial body of research focuses on integrating ML models into the B&B framework [5, 13, 31]. A prominent strand within this area is Learning to Branch (L2B), which aims to learn heuristic policies for selecting branching variables to reduce tree size and solving time [14, 18, 35, 13, 31]. Recent advances include reinforcement learning from suboptimal demonstrations [10] and expressive multi-aggregator graph encoders [33]. In parallel, another line of work leverages ML to predict high-quality solutions directly, often combined with a subsequent search procedure (Predict and Search, PaS). These methods learn to estimate solution distributions or generate initial assignments, which are then used to warm-start or guide a conventional solver for further refinement [32, 21, 34]. Recent developments include iterative prediction–correction frameworks [28] and unsupervised differentiable solving [16]. Together, these two directions represent some of the most promising paradigms for enhancing MILP solvers with ML.

**Contrastive Learning for MILP.** Contrastive learning has been recently adapted to learn MILP representations, primarily to improve data efficiency. In the context of branching, CAMBranch [27] generates positive pairs by applying a simple variable shift to an instance, creating an augmented MILP that shares an identical branching strategy with the original. However, this transformation leaves the core combinatorial structure largely unchanged and does not address the inherent sensitivity of MILPs to more meaningful perturbations. For solution prediction, a notably distinct approach is taken by ConPaS [23], which applies CL directly to solutions within a given instance. It constructs positive and negative samples from high-quality and low-quality (or infeasible) solutions, respectively, to learn a discriminative solution predictor. Consequently, while CAMBranch contrasts problem instances, ConPaS contrasts solutions within a given instance.

**MILP Instance Generation.** A complementary line of work focuses on generating synthetic MILP instances to expand or diversify training distributions, including deep generative models [15, 17, 29]. These methods produce standalone instances and do not, by design, assign generated instances to anchors or provide anchor-relative invariance guarantees. In contrast, MIRAGE constructs anchor-conditioned contrastive pairs with explicit solver-level invariance semantics: each augmented instance is derived from a specific anchor via certificate-constrained perturbations that preserve either the B&B tree structure or the optimality of the nominal solution. The two approaches are therefore complementary: generative methods may enlarge the anchor pool, after which MIRAGE can construct certificate-constrained pairs around those anchors.

## 3 Preliminaries

**Mixed-Integer Linear Programming (MILP).** The MILP is defined as:

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad z = \boldsymbol{c}^\top \boldsymbol{x} \tag{1a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}, \boldsymbol{\ell} \le \boldsymbol{x} \le \boldsymbol{u}, \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{R}^{n-p}. \tag{1b}$$

where $\boldsymbol{x}$ denotes the $n$-dimensional decision variables, consisting of $p$ integer components and $n-p$ continuous variables. The vector $\boldsymbol{c} \in \mathbb{R}^n$ denotes the coefficients of the objective function, $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ is the constraint coefficient matrix, and $\boldsymbol{b} \in \mathbb{R}^m$ represents the right-hand side terms of the constraints. The vectors $\boldsymbol{\ell} \in (\mathbb{R}\bigcup\{-\infty\})^n$ and $\boldsymbol{u} \in (\mathbb{R}\bigcup\{+\infty\})^n$ specify the lower and upper bounds for the decision variables, respectively.

**Branch-and-Bound (B&B).** The branch-and-bound (B&B) framework is a fundamental and widely used methodology for obtaining exact solutions to NP-hard optimization problems [24]. It performs an implicit enumeration of all feasible solutions by organizing partial solutions, referred to as subproblems, in a search tree. At each unexplored node, the algorithm generates child nodes by partitioning the solution space into smaller regions that are solved recursively (i.e., branching), while bounding rules are applied to prune regions that are provably suboptimal. After the entire tree has been processed, the best feasible solution identified during the search is returned.

# 4 Methodology

In this section, we build MIRAGE around a single, solver-recorded quantity: the piecewise-affine B&B dual price function $F_{\mathrm{BB}}$ induced by an LP-based branch-and-bound run (Sec. 4.1). We first show that $F_{\mathrm{BB}}$ acts as an affine probe of instance perturbations, so that right-hand-side (RHS) and objective shifts enter it through the recorded dual vectors. This yields a unifying invariance principle from which our two certificates follow as two projections: RHS perturbations that preserve the search-tree geometry (Sec. 4.2) and objective perturbations that preserve the nominal optimality (Sec. 4.3). Finally, we instantiate this principle as a data-augmentation mechanism whose certified perturbations serve as positives and whose $F_{\mathrm{BB}}$-shifting perturbations serve as negatives (Sec. 4.4).

## 4.1 B&B dual price function

We leverage the piecewise-affine dual price function $F_{\mathrm{BB}}$ induced by the branch-and-bound algorithm, which provides a strong dual for MILPs and is readily recorded during a solver run [20]. The central observation is that $F_{\mathrm{BB}}$ is a affine probe of instance perturbations: right-hand-side shifts $\Delta b$ and objective shifts $\Delta c$ enter $F_{\mathrm{BB}}$ through the recorded dual vectors $(\lambda^t)$ and $(\mu^t, \bar{\mu}^t)$ respectively (detailed in Appendix A.1). This motivates a unifying principle: perturbations that leave $F_{\mathrm{BB}}$ invariant on the recorded B&B tree preserve either the search-tree geometry or the nominal optimality. The two certificates below (in Sec. 4.2 and 4.3) are therefore not independent constructions but two projections of the same invariance principle: they characterize perturbation directions along which every component $\phi_t$ of $F_{\mathrm{BB}}$ — and hence $F_{\mathrm{BB}}$ itself — is preserved on the recorded B&B tree. In contrast, random-sign perturbations (used as negatives in Sec. 4.4) provably shift $F_{\mathrm{BB}}$ by a non-trivial margin (Appendix A.5), giving a principled separation between positives and negatives measured in a single, B&B-induced dual quantity.

## 4.2 RHS perturbations: a computable inactive-basic certificate

We first project the invariance principle onto right-hand-side space. The goal is to isolate a computable set of RHS shifts along which every recorded node LP retains its optimal basis, so that the search-tree geometry — and hence $F_{\mathrm{BB}}$ on the recorded prefix — is preserved.

Let the algorithm solve Eq. 1 at the nominal RHS $\boldsymbol{b}_0$. For each recorded node $t$, let $B^t$ be the selected optimal simplex basis of the node LP Eq. 15 at $\boldsymbol{b}_0$ after introducing slack variables

$$\boldsymbol{A}\boldsymbol{x} + \boldsymbol{\sigma} = \boldsymbol{b},\ \boldsymbol{\sigma} \geq \boldsymbol{0}. \tag{2}$$

Let $\mathcal{T}_{\leq K}$ denote the recorded nodes up to depth $K$, and let $\boldsymbol{x}^t$ be the selected optimal basic solution at node $t$. The slack of original constraint $i$ at node $t$ is

$$s_i^t := b_{0,i} - \boldsymbol{a}_i^\top \boldsymbol{x}^t,\ i = 1, \ldots, m, \tag{3}$$

where $\boldsymbol{a}_i^\top$ is the $i$-th row of $\boldsymbol{A}$.

For a tolerance $\tau > 0$, define the set of constraints that are uniformly inactive and basic throughout the recorded prefix:

$$\mathcal{I}_{K,\tau} := \left\{ i \in \{1, \ldots, m\} : s_i^t \geq \tau \text{ and } \sigma_i \text{ is basic in } B^t,\ \forall t \in \mathcal{T}_{\leq K} \right\}. \tag{4}$$

For each $i \in \mathcal{I}_{K,\tau}$, let

$$\underline{s}_i := \min_{t \in \mathcal{T}_{\leq K}} s_i^t. \tag{5}$$

By construction, $\underline{s}_i \geq \tau$. For $0 < \alpha < 1$, define the certified RHS perturbation region

$$\mathcal{Q}_{K,\tau,\alpha} := \{ \Delta\boldsymbol{b} \in \mathbb{R}^m : \Delta b_i = 0\ \forall i \notin \mathcal{I}_{K,\tau},\ |\Delta b_i| \leq \alpha \underline{s}_i\ \forall i \in \mathcal{I}_{K,\tau} \}. \tag{6}$$

This region is a conservative, computable subset of RHS perturbations supported only on constraints whose slacks are basic and uniformly positive on the recorded prefix.

**Theorem 4.1** (Inactive-basic RHS certificate)**.** *For any $\Delta\boldsymbol{b} \in \mathcal{Q}_{K,\tau,\alpha}$ and any node $t \in \mathcal{T}_{\leq K}$, the same basis $B^t$ remains primal feasible and optimal for the perturbed node LP with RHS $\boldsymbol{b}_0 + \Delta\boldsymbol{b}$. Moreover, the decision-variable component of the selected basic solution is unchanged:*

$$\boldsymbol{x}^t(\boldsymbol{b}_0 + \Delta\boldsymbol{b}) = \boldsymbol{x}^t(\boldsymbol{b}_0). \tag{7}$$

*Consequently,*

$$z_t(\boldsymbol{b}_0 + \Delta\boldsymbol{b}) = z_t(\boldsymbol{b}_0). \tag{8}$$

*Proof.* Shown in the Appendix A.2. □

As a direct consequence, $\phi_t(b_0 + \Delta b) = \phi_t(b_0)$ for all recorded $t$, and therefore $F_{\mathrm{BB}}$ is invariant on the recorded prefix under any $\Delta b \in \mathcal{Q}_{K,\tau,\alpha}$. The certificate above does not attempt to characterize the full solver-dependent stability region. Under deterministic solver abstraction, Theorem 4.1 gives a sufficient condition for preserving the recorded B&B prefix, i.e., the search-tree geometry underlying $F_{\mathrm{BB}}$. We state the complete prefix-stability implication and its assumptions in Appendix A.3.

### 4.3 Objective perturbations: optimality and value certificates

We now project the same principle onto objective space. Here the target invariant is nominal optimality: we characterize objective shifts under which a nominal optimal solution $\boldsymbol{x}^\star$ stays optimal over the same feasible set, so that $F_{\mathrm{BB}}$ is preserved through its recorded dual price.

Fix $\boldsymbol{b} = \boldsymbol{b}_0$ and let $\mathcal{X}_I$ be the feasible set. Let $\boldsymbol{x}^\star \in \mathcal{X}_I$ be any optimal solution of the nominal MILP with objective coefficient $\boldsymbol{c}$. The following certificate does not require $\boldsymbol{x}^\star$ to be unique and does not rely on a positive global optimality gap.

Define the index sets

$$\begin{aligned} J_L &:= \{j : \ x_j^\star = \ell_j < u_j\}, \ J_U := \{j : \ \ell_j < x_j^\star = u_j\}, \\ J_F &:= \{j : \ \ell_j < x_j^\star < u_j\}, J_{\mathrm{fix}} := \{j : \ \ell_j = x_j^\star = u_j\}. \end{aligned} \tag{9}$$

For a minimization problem, define the box-normal objective cone at $\boldsymbol{x}^\star$:

$$\mathcal{C}(\boldsymbol{x}^\star) := \{\Delta \boldsymbol{c} \in \mathbb{R}^n : \Delta c_j \geq 0 \ \forall j \in J_L, \ \Delta c_j \leq 0 \ \forall j \in J_U, \ \Delta c_j = 0 \ \forall j \in J_F\}, \tag{10}$$

with no sign restriction on $J_{\mathrm{fix}}$, since fixed variables do not change over $\mathcal{X}_I$. For maximization models, we first convert the problem to minimization.

**Theorem 4.2** (Objective optimality certificate)**.** *For any $\Delta \boldsymbol{c} \in \mathcal{C}(\boldsymbol{x}^\star)$, the solution $\boldsymbol{x}^\star$ remains optimal for the perturbed objective $\boldsymbol{c} + \Delta \boldsymbol{c}$ over the same feasible set $\mathcal{X}_I$. Moreover,*

$$z(\boldsymbol{c} + \Delta \boldsymbol{c}, \boldsymbol{b}_0) = (\boldsymbol{c} + \Delta \boldsymbol{c})^\top \boldsymbol{x}^\star = z(\boldsymbol{c}, \boldsymbol{b}_0) + \Delta \boldsymbol{c}^\top \boldsymbol{x}^\star. \tag{11}$$

*Proof.* Shown in the Appendix A.4. □

A useful value-invariant special case is to perturb only variables that are zero at $\boldsymbol{x}^\star$ and have zero lower bound: $J_0 := \{j : \ x_j^\star = 0, \ \ell_j = 0\}$.

If

$$\Delta c_j \geq 0 \ \forall j \in J_0, \ \Delta c_j = 0 \ \forall j \notin J_0, \tag{12}$$

then $\Delta \boldsymbol{c} \in \mathcal{C}(\boldsymbol{x}^\star)$ and $\Delta \boldsymbol{c}^\top \boldsymbol{x}^\star = 0$. Therefore, by Theorem 4.2,

$$z(\boldsymbol{c} + \Delta \boldsymbol{c}, \boldsymbol{b}_0) = z(\boldsymbol{c}, \boldsymbol{b}_0). \tag{13}$$

In particular, under this case, $\phi_t$ evaluated at the nominal RHS is unchanged for the root node certificate, so $F_{\mathrm{BB}}(b_0)$ is preserved — the objective certificate is the $\Delta c$-counterpart of the RHS certificate under the same dual-price invariance lens.

### 4.4 Augmentation-Based Representation Learning

Building upon the theoretical foundation above, we present MIRAGE, a contrastive learning framework for MILP representation learning based on augmentation. This framework first generates positive samples through two distinct augmentation schemes and subsequently performs contrastive learning. Corresponding to the two augmentation strategies derived in Sec 4.2 and Sec. 4.3, our approach comprises two specific variants: MIRAGE-C (based on constraint perturbation) and MIRAGE-O (based on objective perturbation).

For MIRAGE-C, positive samples are generated by perturbing the RHS $\boldsymbol{b}$ within the inactive-basic certificate region $\mathcal{Q}_{K,\tau,\alpha}$ established in Theorem 4.1: the perturbation support is restricted to constraints whose slacks are uniformly basic and strictly positive throughout the recorded B&B

Table 1: Performance comparison of branching policies across four MILP benchmarks, evaluated using the 1-shifted geometric mean of solving time (Time), number of branching nodes (Nodes), and number of wins (Wins). Further experimental details are provided in Sec. 5.1. Values highlighted in bold indicate where the proposed method achieves the best performance among all baselines.

| **SC** | Easy | | | Medium | | | Hard | | |
|---|---|---|---|---|---|---|---|---|---|
| Model | Time↓ | Wins↑ | Nodes↓ | Time↓ | Wins↑ | Nodes↓ | Time↓ | Wins↑ | Nodes↓ |
| FSB | 19.52 | 0/100 | 24 | 452.69 | 0/88 | 206 | 3600 | 0/0 | - |
| RPB | 9.74 | 1/100 | 108 | 71.84 | 3/100 | 2147 | 1948.62 | 10/62 | 92766 |
| GCNN [14] | 8.79 | 46/100 | 201 | 55.80 | 47/100 | 2108 | 1897.31 | 26/66 | 90490 |
| CAMBranch [27] | 9.04 | 2/100 | 206 | 60.46 | 8/100 | 2276 | 2022.92 | 1/57 | 108204 |
| MIRAGE-GCNN | **8.57** | **51/100** | **187** | **55.47** | 42/100 | 2142 | **1876.78** | **33/65** | 90812 |
| **CA** | Easy | | | Medium | | | Hard | | |
| Model | Time↓ | Wins↑ | Nodes↓ | Time↓ | Wins↑ | Nodes↓ | Time↓ | Wins↑ | Nodes↓ |
| FSB | 5.28 | 0/100 | 12 | 88.16 | 0/100 | 95 | 1914.65 | 0/69 | 427 |
| RPB | 2.94 | 4/100 | 47 | 18.71 | 1/100 | 676 | 145.28 | 4/100 | 8250 |
| GCNN [14] | 1.99 | 17/100 | 117 | 11.63 | 37/100 | 749 | 124.77 | 17/100 | 13197 |
| CAMBranch [27] | 1.80 | 26/100 | 109 | 12.97 | 16/100 | 781 | 117.53 | 14/100 | 11036 |
| MIRAGE-GCNN | **1.72** | **53/100** | **104** | **11.27** | **46/100** | **702** | **106.91** | **75/100** | **9027** |
| **CFL** | Easy | | | Medium | | | Hard | | |
| Model | Time↓ | Wins↑ | Nodes↓ | Time↓ | Wins↑ | Nodes↓ | Time↓ | Wins↑ | Nodes↓ |
| FSB | 37.34 | 3/100 | 17 | 285.02 | 0/100 | 142 | 1089.39 | 1/77 | 107 |
| RPB | 32.03 | 11/100 | 46 | 148.94 | 10/100 | 208 | 805.23 | 9/93 | 224 |
| GCNN [14] | 28.86 | 24/100 | 122 | 138.60 | 17/100 | 373 | 714.27 | 13/91 | 501 |
| CAMBranch [27] | 27.55 | 26/100 | 104 | 135.11 | 41/100 | 344 | 642.75 | 20/94 | 430 |
| MIRAGE-GCNN | **26.90** | **36/100** | **98** | 136.89 | 32/100 | 357 | **627.84** | **51/94** | 452 |
| **MIS** | Easy | | | Medium | | | Hard | | |
| Model | Time↓ | Wins↑ | Nodes↓ | Time↓ | Wins↑ | Nodes↓ | Time↓ | Wins↑ | Nodes↓ |
| FSB | 31.56 | 0/100 | 22 | 1763.21 | 0/51 | 96 | 3600 | 0/0 | - |
| RPB | 11.45 | 8/100 | 125 | 169.32 | 4/100 | 4168 | 1041.43 | 5/36 | 29832 |
| GCNN [14] | 7.48 | 16/100 | 177 | 195.87 | 1/80 | 6236 | 1367.06 | 0/25 | 30249 |
| CAMBranch [27] | 6.96 | 29/100 | 150 | 137.16 | 27/100 | 5182 | 863.19 | 7/45 | 22643 |
| MIRAGE-GCNN | **6.88** | **47/100** | 156 | **122.63** | **68/100** | **4863** | **732.29** | **40/47** | 23150 |

prefix, and $|\Delta b_i| \leq \alpha s_i$. By Theorem 4.1, such perturbations leave every node-LP solution and value on the recorded prefix *exactly unchanged*, so the resulting instance shares the same branching trajectory with the anchor and constitutes a structurally valid positive sample. Negative samples are produced by drawing random-sign perturbations on the full constraint set with the same $\|\Delta \boldsymbol{b}\|$ as the positives; these perturbations lie outside $\mathcal{Q}_{K,\tau,\alpha}$ and, by the drift analysis in Appendix A.5, induce non-trivial changes in both the node-LP values and the induced dual price function $F_{\mathrm{BB}}$.

For MIRAGE-O, positive samples are generated by perturbing the objective coefficients $\boldsymbol{c}$ inside the box-normal cone $\mathcal{C}(\boldsymbol{x}^\star)$ at a nominal optimum $\boldsymbol{x}^\star$ (Theorem 4.2). We focus on the value-invariant special case, which guarantees both that $\boldsymbol{x}^\star$ remains optimal under $\boldsymbol{c} + \Delta\boldsymbol{c}$ and that $z(\boldsymbol{c} + \Delta\boldsymbol{c}, \boldsymbol{b}_0) = z(\boldsymbol{c}, \boldsymbol{b}_0)$. Negative samples are drawn as random-sign perturbations of $\boldsymbol{c}$ on the full support, matched in magnitude $\|\Delta\boldsymbol{c}\|$; by the Khintchine bound (Appendix A.5), such perturbations incur an expected absolute drift $\mathbb{E}|\Delta\boldsymbol{c}^\top \boldsymbol{x}^\star| \geq \kappa\rho\|\boldsymbol{x}^\star_S\|_2$, breaking value invariance and providing a principled contrastive counterpart.

As a representation learning framework, MIRAGE is designed to be readily incorporated into the pipelines of other ML methods for solving MILPs. It functions to learn high-quality representations that capture task-salient instance features, thereby providing an informed parameter initialization for downstream ML models. Specifically, before task-specific training, the model first processes training instances along with their generated positive and negative samples (which may require minor architectural adjustments) to compute their representations. Then it is trained through the classic contrastive loss function [8, 22], defined as

$$\mathcal{L} = -\sum_{i=1}^{B} \log \frac{\exp(\mathrm{sim}_i^+ / \tau_{\mathrm{temp}})}{\exp(\mathrm{sim}_i^+ / \tau_{\mathrm{temp}}) + \sum_{j=1}^{N} \exp(\mathrm{sim}_{i,j}^- / \tau_{\mathrm{temp}})}, \tag{14}$$

where $B$ is the batch size, $\mathrm{sim}_i^+$ denotes the cosine similarity between the representation of the $i$-th instance and its positive sample, $N$ is the number of negative samples, $\mathrm{sim}_{i,j}^-$ is the cosine similarity between the $i$-th instance and its $j$-th negative sample, and $\tau_{\mathrm{temp}}$ is a temperature hyperparameter.

Table 2: Performance comparison of Predict-and-Search across four MILP benchmarks, evaluated by the average objective values (OBJ) and absolute primal gap ($\text{gap}_{\text{abs}}$) under a 1,000s time limit. Results are shown separately for Gurobi and SCIP as backend solvers. Further experimental details are provided in Sec. 5.2. Values highlighted in bold denote the best performance among all baselines.

| | IP (BKS: 11.89) | | WA (BKS: 708.19) | | CA (BKS: 97456.38) | | MIS (BKS: 913.06) | |
|---|---|---|---|---|---|---|---|---|
| | OBJ↓ | $\text{gap}_{\text{abs}}$↓ | OBJ↓ | $\text{gap}_{\text{abs}}$↓ | OBJ↑ | $\text{gap}_{\text{abs}}$↓ | OBJ↑ | $\text{gap}_{\text{abs}}$↓ |
| Gurobi | 12.49 | 0.60 | 708.96 | 0.77 | 97132.76 | 323.62 | 913.03 | 0.03 |
| PS+Gurobi | 12.41 | 0.52 | 708.88 | 0.69 | 97195.27 | 261.11 | 913.05 | 0.01 |
| ConPaS+Gurobi | 12.32 | 0.43 | 708.83 | 0.64 | 97247.13 | 209.25 | **913.06** | **0.00** |
| MIRAGE-PS+Gurobi | **12.26** | **0.37** | **708.71** | **0.52** | **97269.85** | **186.53** | **913.06** | **0.00** |
| Improvement over PS | | 28.8% | | 24.6% | | 28.6% | | 100.0% |
| SCIP | 14.28 | 2.39 | 709.66 | 1.47 | 96302.39 | 1153.99 | 912.94 | 0.12 |
| PS+SCIP | 14.19 | 2.30 | 709.43 | 1.24 | 96460.47 | 995.91 | 913.01 | 0.05 |
| ConPaS+SCIP | 13.85 | 1.96 | 709.21 | 1.02 | 96568.42 | 887.96 | 913.05 | 0.01 |
| MIRAGE-PS+SCIP | **13.71** | **1.82** | **709.07** | **0.88** | **96637.18** | **819.20** | **913.06** | **0.00** |
| Improvement over PS | | 20.9% | | 29.0% | | 17.7% | | 100.0% |

Regarding computational cost, on an evaluated benchmark, dual recording piggybacks on the routine solver run required for label generation, adding roughly 1.8% wall-clock time; the one-time contrastive pre-training overhead is roughly 20 minutes with 3.2/6.4 GB peak GPU/CPU memory, and MIRAGE introduces no additional test-time operations (detailed in Appendix D).

# 5 Experiments

To evaluate the effectiveness of MIRAGE, we conduct comprehensive experiments on two prominent MILP tasks: Learning to Branch (L2B) and Predict-and-Search (PaS). We integrate MIRAGE with existing ML methods to enhance their learned representations. Specifically, we leverage MIRAGE-C for L2B, as it preserves the B&B search tree structure, and adopt MIRAGE-O for PaS, given its property of retaining solution feasibility and optimality. Furthermore, we visualize the representations learned by MIRAGE to offer additional insight into its performance.

## 5.1 Experimental Results on Learning to Branch

**Datasets.** Following the established benchmark in [14], we evaluate our approach on four NP-hard problems: Set Covering (SC) [4], Combinatorial Auction (CA) [26], Capacitated Facility Location (CFL) [9], and Maximum Independent Set (MIS) [6]. Each benchmark includes instances of three difficulty levels: *Easy*, *Medium*, and *Hard*. We employ SCIP 7.0.3 [12] as the backend solver with a one-hour time limit. Further details are provided in Appendix B.1.

**Baselines.** We integrate our approach with GCNN [14], denoted as **MIRAGE-GCNN**. It is compared against the following baselines: 1) Reliability Pseudocost Branching (RPB) [2], the state-of-the-art expert-designed default policy in SCIP; 2) GCNN [14], a widely-adopted neural branching policy; 3) CAMBranch [27], a competing method that leverages contrastive learning in a different manner. All ML methods (GCNN, CAMBranch, MIRAGE-GCNN) adopt the identical network architecture to ensure fair comparison.

**Training Setup.** In line with [14], all models are trained on *Easy*-level instances and evaluated across all difficulty levels, with separate models trained for each problem. The original training set for each problem comprises 10,000 randomly generated instances. From this set, we randomly select 1,000 instances and use MIRAGE-C to generate 1,000 corresponding positive samples. The number of negative samples per instance, $n_{\text{neg}}$, is set to 10, yielding a total of 10,000 negative samples. To ensure a fair comparison, the generated positive samples are incorporated into the original training set. Since the negative samples are not guaranteed to be feasible, they are used exclusively for the contrastive pre-training phase in our MIRAGE-GCNN. Consequently, for GCNN and CAMBranch, training is performed on 100,000 expert branching samples extracted from 11,000 instances, with a separate 20,000 samples from 2,000 instances held out for validation. For our MIRAGE-GCNN, we first perform contrastive learning on the 1,000 selected instances along with their positive and negative samples, after which the subsequent imitation learning procedure is kept identical to GCNN.

**Evaluation.** Models are evaluated across difficulty levels, each on 20 new instances using 5 different seeds, resulting in a total of 100 solving attempts. We report standard metrics for MILP benchmarking: the solving time in seconds (Time), the number of branching nodes (Nodes), and the number of times each policy achieves the fastest solving time, over the number of solved instances (Wins).

Table 3: Pooled ablation results, showing mean relative improvement with 95% confidence intervals across 12 L2B configurations (SC, CA, CFL, and MIS, each at Easy/Medium/Hard difficulty levels) and 6 PaS settings (IP, WA, and CA, each solved with Gurobi and SCIP).

| Comparison | L2B Time Improvement | PaS $\text{gap}_{\text{abs}}$ Improvement |
|---|---|---|
| A1-A0: certified data alone | 2.3% [1.1, 3.6] | 4.9% [2.8, 7.0] |
| A2-A0: certified CL alone | 12.4% [8.0, 16.7] | 25.3% [20.0, 30.3] |
| A3-A1: CL on identical data | 12.3% [8.0, 16.5] | 24.9% [19.5, 30.0] |
| A3-A2: added certified data | 2.4% [1.0, 3.8] | 4.5% [2.1, 6.8] |
| **A3-A4: certificate vs random** | **12.3% [8.1, 16.4]** | **24.9% [19.7, 29.8]** |
| A4-A1: random-positive CL | 0.0% [-1.4, 1.5] | 0.1% [-1.8, 2.0] |

**Results.** The experimental results, summarized in Table 1, demonstrate that ML-based branching policies consistently surpass human-designed heuristics, affirming the potential of data-driven approaches for this task. We further observe that CAMBranch, which also serves as a data augmentation tool, generally outperforms GCNN when both are trained on the complete dataset, with the exception of the SC benchmark. Crucially, our approach exceeds the performance of CAMBranch on three of the four benchmarks and delivers comparable results on the CFL problem. These findings suggest that our approach, by moving beyond simple variable shifts, provides a more powerful means of data augmentation, thereby enabling the learning of better representations and branching policies.

### 5.2 Experimental Results on Predict-and-Search

**Datasets.** Following the experimental protocol of [21], we conduct experiments on four popular MILP benchmarks for PaS. These include two synthetic problems—Combinatorial Auction (CA) [26] and Maximum Independent Set (MIS) [6]—alongside two challenging real-world benchmarks from the NeurIPS ML4CO 2021 competition [13]: Item Placement (IP) and Workload Appointment (WA). We employ SCIP 7.0.3 [12] and Gurobi 9.5.2 [19] as backend solvers for comprehensive evaluation. Further details are provided in Appendix C.1.

**Baselines.** We integrate our approach with PS [21], denoted as **MIRAGE-PS**. Our evaluation compares against the following baselines: 1) PS [21], which predicts variable assignments followed by a trust-region search performed by MILP solvers; 2) ConPaS [23], which enhances PS through contrastive learning between high- and low-quality solutions within individual instances. For ConPaS, we adhere to the experimental setup in [23] with a positive-to-negative sample ratio of 10, using low-quality solutions as negative samples. An identical network architecture is adopted for all ML methods. Additionally, we include the standalone performance of both backend solvers—configured to prioritize primal solution discovery—to provide comprehensive comparison.

**Training Setup.** All models are trained separately on each dataset. In line with [21], we partition the data into 240 instances for training, 60 for validation, and 100 for testing. From the original training set, we randomly select 60 instances and apply MIRAGE-O to generate 60 corresponding positive samples. The number of negative samples per instance, $n_{\text{neg}}$, remains 10, yielding a total of 600 negative samples. Mirroring the setup in our L2B experiments, the generated positive samples are incorporated into the original training set. Although the negative samples are feasible, we empirically found that including them in training baselines induces performance degradation, probably due to a significant shift in the problem distribution. Consequently, PS and ConPaS are trained on the set of 300 instances (240 original + 60 positive). For MIRAGE-PS, we first perform contrastive learning on the 60 instances along with their associated positive and negative samples, after which the subsequent training procedure is identical to PS.

**Evaluation.** Methods are evaluated on each test instance by recording the best objective value (OBJ) obtained within a 1,000-second time limit. To establish a reference for solution quality, we compute a best-known solution (BKS) by running single-threaded Gurobi with an extended 3,600-second time limit. Performance is quantified using the absolute primal gap, defined as $\text{gap}_{\text{abs}} := |\text{OBJ} - \text{BKS}|$, where smaller values indicate superior performance. Additionally, we report the relative improvement of our approach over PS, calculated as $(|\text{gap}^{\text{PS}}_{\text{abs}} - \text{gap}^{\text{Ours}}_{\text{abs}}|)/\text{gap}^{\text{PS}}_{\text{abs}}$, to directly demonstrate the benefit of our data augmentation strategy.

**Results.** As summarized in Table 2, ML-enhanced methods consistently surpass the performance of standalone solvers, affirming the significant potential of the PaS paradigm. Our proposed method

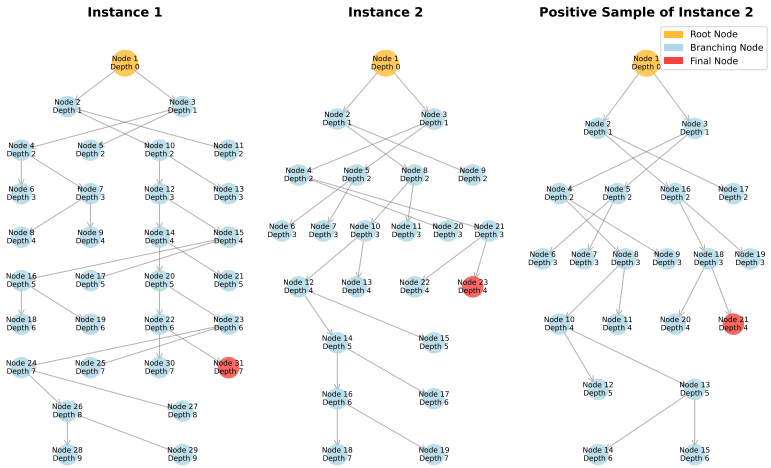



Figure 1: Visualization of B&B search trees. Each vertex represents a branching node, annotated with its exploration order and depth. Instance 1 and instance 2 are different instances from the same source (combinatorial auction). The positive sample of instance 2 is generated via the augmentation strategy of MIRAGE-C.

achieves a substantial reduction in the absolute primal gap over PS across all benchmarks and backend solvers. This validates the efficacy of our instance augmentation strategy in learning superior representations for PaS. Furthermore, our approach also maintains a clear performance advantage over ConPaS, which applies contrastive learning at the solution level. This comparative result highlights that our instance-level contrastive paradigm, which learns from structural variations of the problem itself, provides a more effective inductive bias than contrasting solutions within a fixed instance. The consistent superiority of our method underscores the inherent advantage of capturing foundational problem structures via instance augmentation.

### 5.3 Ablation Studies

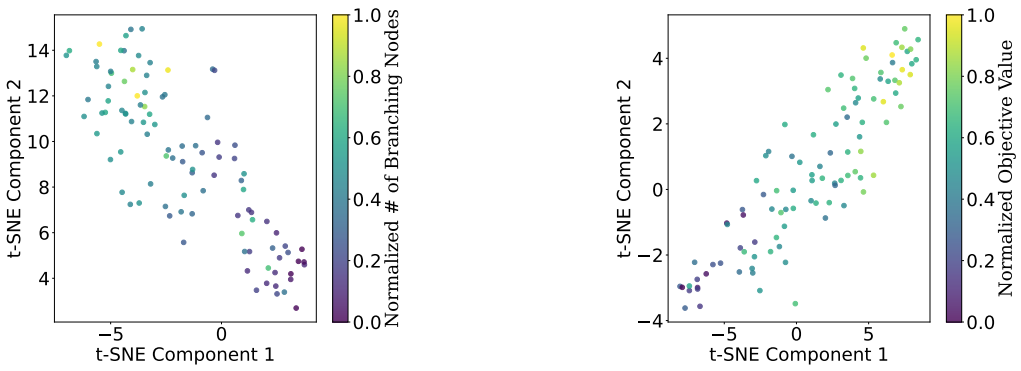
To isolate MIRAGE's contribution and validate the role of certificate-constrained positives, we conduct controlled ablations.

**Does the certificate matter? A five-arm factorial ablation.** We design a 2×2 factorial ablation with an additional certificate-isolation arm to separate the effects of (i) certificate-generated task data and (ii) certificate-constrained contrastive pre-training:

- **A0 (Base)**: Original task-training data; no contrastive pre-training.
- **A1 (CertData-NoCL)**: Original + certificate-generated task-training data; no CL.
- **A2 (CertCL-Only)**: Original task-training data only; certificate-constrained CL.
- **A3 (MIRAGE-Full)**: Certificate-generated task-training data + certificate-constrained CL.
- **A4 (RandPos-CL)**: Identical to A3, except certified positives are replaced with support-cardinality- and norm-matched non-certified positives; negatives and downstream task data remain unchanged.

GCNN/PS arms correspond to A1: they use the same certified task data as MIRAGE but omit CL. Thus, A3 - A1 isolates CL under identical downstream data, while A3 - A4 isolates the certificate-constrained positive construction. For A4, each certified positive is replaced with a sparse non-certified positive: we sample the same number of perturbed coordinates, preserve the per-anchor norm, and assign random signs. Draws that remain inside the corresponding certificate region are rejected; infeasible MIRAGE-C draws are also rejected. Negatives remain the original full-support, norm-matched random-sign perturbations. Problem scale, architecture, initialization, optimization steps, and task data are otherwise identical. The results are shown in Table 3.

We conclude that Certified CL is the dominant factor (12.4%/25.3% improvement without augmented task data), while Random-positive CL is indistinguishable from A1. Hence, neither extra data nor




(a) MIRAGE-C (b) MIRAGE-O

Figure 2: t-SNE visualization of MILP instance representations. Each point corresponds to a single instance. (a) Representations colored by the normalized number of branching nodes, produced by the model trained with the MIRAGE-C strategy. (b) Representations colored by the normalized objective value, produced by the model trained with the MIRAGE-O strategy.

random-positive CL explains the gain; the dominant improvement is associated with certificate-constrained positives.

### 5.4 Visualizations

**B&B Search Tree.** Fig. 1 visualizes the B&B search tree structures for two different instances from the Combinatorial Auction benchmark and a positive sample generated via MIRAGE-C. Each vertex represents a branching node, annotated with its exploration order and depth. While different instances from the same problem source can exhibit significant structural differences in their corresponding B&B trees, the positive sample of Instance 2 (right) largely preserves the tree topology of its anchor (middle). Negative samples (not shown) often result in markedly simplified trees, typically containing only one or two nodes due to infeasibility or a collapse in problem-solving characteristics.

**Representation Space.** Fig 2 presents t-SNE visualizations of MILP instance representations after contrastive pre-training. (a) MIRAGE-C: embeddings colored by the normalized number of branching nodes, revealing a clear gradient where instances requiring fewer nodes cluster in the bottom-right and more complex instances populate the top-left. (b) MIRAGE-O: embeddings colored by the normalized objective value, with larger values concentrating in the top-right and smaller ones in the bottom-left. These results demonstrate that our augmentation and contrastive pre-training strategy effectively encodes meaningful structural and objective features of MILP instances into the learned representation space.

## 6 Conclusion

In this paper, we present MIRAGE, a duality-inspired representation learning framework for MILPs built on a single unifying principle: invariance of the B&B dual price function on the recorded search tree. From this principle, we derive two complementary augmentation certificates—an inactive-basic RHS certificate (MIRAGE-C) and a box-normal objective certificate (MIRAGE-O)—as computable, instance-specific projections that convert recorded B&B basis/slack information or a nominal optimizer into executable positive regions for MILP contrastive learning. Under deterministic solver assumptions, perturbations within these regions provably preserve either the B&B tree structure (K-prefix identity) or the optimality of the nominal solution, with empirical preservation rates of 96.9% and 100% under standard SCIP, respectively. Controlled ablations confirm that certificate-constrained positives outperform support-cardinality- and norm-matched non-certified positives in the evaluated settings. Empirical integration with ML methods for L2B and PaS demonstrates consistent performance gains. By grounding MILP data augmentation in a single, solver-induced duality quantity, our work provides a principled and effective approach to MILP representation learning.

## References


[1] Tobias Achterberg. Scip: solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[2] Tobias Achterberg, Thorsten Koch, and Alexander Martin. Branching rules revisited. *Operations Research Letters*, 33(1):42–54, 2005.

[3] Achim Bachem and Walter Kern. *Linear Programming Duality: An Introduction to Oriented Matroids*. Springer, 1992.

[4] Egon Balas and Andrew Ho. Set covering algorithms using cutting planes, heuristics, and subgradient optimization: a computational study. In *Combinatorial optimization*, pages 37–60. Springer, 2009.

[5] Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, 290(2):405–421, 2021.

[6] David Bergman, Andre A Cire, Willem-Jan Van Hoeve, and John Hooker. *Decision diagrams for optimization*. Springer, 2016.

[7] Nadjib Brahimi and Tarik Aouam. Multi-item production routing problem with backordering: a milp approach. *International Journal of Production Research*, 54(4):1076–1093, 2016.

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[9] Gérard Cornuéjols, Ranjani Sridharan, and Jean-Michel Thizy. A comparison of heuristics and relaxations for the capacitated plant location problem. *European journal of operational research*, 50(3):280–297, 1991.

[10] Shengyu Feng and Yiming Yang. Sorrel: suboptimal-demonstration-guided reinforcement learning for learning to branch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 11212–11220, 2025.

[11] Christodoulos A Floudas and Xiaoxia Lin. Mixed integer linear programming in process scheduling: Modeling, algorithms, and applications. *Annals of Operations Research*, 139(1):131–162, 2005.

[12] Gerald Gamrath, Daniel Anderson, Ksenia Bestuzheva, Wei-Kun Chen, Leon Eifler, Maxime Gasse, Patrick Gemander, Ambros Gleixner, Leona Gottwald, Katrin Halbig, et al. The scip optimization suite 7.0. Technical report, 2020.

[13] 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*, volume 176, pages 220–231. PMLR, 2022.

[14] Maxime Gasse, Didier Chételat, Nicola Ferroni, Laurent Charlin, and Andrea Lodi. Exact combinatorial optimization with graph convolutional neural networks. volume 32, 2019.

[15] 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. *Advances in Neural Information Processing Systems*, 36:26025–26047, 2023.

[16] Zijie Geng, Jie Wang, Xijun Li, Fangzhou Zhu, Jianye Hao, Bin Li, and Feng Wu. Differentiable integer linear programming. In *International Conference on Learning Representations*, volume 2025, pages 28283–28309, 2025.

[17] Ziao Guo, Yang Li, Chang Liu, Wenli Ouyang, and Junchi Yan. Acm-milp: Adaptive constraint modification via grouping and selection for hardness-preserving milp instance generation. In *Forty-first International Conference on Machine Learning*, 2024.

[18] Prateek Gupta, Maxime Gasse, Elias Khalil, Pawan Mudigonda, Andrea Lodi, and Yoshua Bengio. Hybrid models for learning to branch. volume 33, pages 18087–18097, 2020.

[19] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024.

[20] Menal Guzelsoy and Theodore K Ralphs. Duality for mixed-integer linear programs. *International Journal of Operations Research*, 4(3):118–137, 2007.

[21] 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.

[22] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[23] Taoan Huang, Aaron M Ferber, Arman Zharmagambetov, Yuandong Tian, and Bistra Dilkina. Contrastive predict-and-search for mixed integer linear programs. In *Forty-first International Conference on Machine Learning*, 2024.

[24] Ailsa H Land and Alison G Doig. An automatic method for solving discrete programming problems. In *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*, pages 105–132. Springer, 2009.

[25] Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. Contrastive representation learning: A framework and review. *Ieee Access*, 8:193907–193934, 2020.

[26] 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.

[27] Jiacheng Lin, Meng Xu, Zhihua Xiong, and Huangang Wang. Cambranch: Contrastive learning with augmented MILPs for branching. In *International Conference on Learning Representations*, 2024.

[28] 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 *International Conference on Learning Representations*, volume 2025, pages 94513–94540, 2025.

[29] Haoyang Liu, Jie Wang, Wanbo Zhang, Zijie Geng, Yufei Kuang, Xijun Li, Yongdong Zhang, Bin Li, and Feng Wu. Milp-studio: Milp instance generation via block structure decomposition. *Advances in Neural Information Processing Systems*, 37:54357–54395, 2024.

[30] Li Liu and Qi Fan. Resource allocation optimization based on mixed integer linear programming in the multi-cloudlet environment. *IEEE Access*, 6:24533–24542, 2018.

[31] Lara Scavuzzo, Karen Aardal, Andrea Lodi, and Neil Yorke-Smith. Machine learning augmented branch and bound for mixed integer linear programming. *Mathematical Programming*, 2024.

[32] Nicolas Sonnerat, Pengming Wang, Ira Ktena, Sergey Bartunov, and Vinod Nair. Learning a large neighborhood search algorithm for mixed integer programs. *arXiv preprint arXiv:2107.10201*, 2021.

[33] Nguyen Vinh Toan and Nguyen Van-Hop. Graph convolutional network with multiple aggregators for learning to branch. *Knowledge-Based Systems*, page 114095, 2025.

[34] Huigen Ye, Hua Xu, Hongyan Wang, Chengming Wang, and Yu Jiang. Gnn&gbdt-guided fast optimizing framework for large-scale integer programming. In *International conference on machine learning*, volume 202, pages 39864–39878. PMLR, 2023.

[35] Giulia Zarpellon, Jason Jo, Andrea Lodi, and Yoshua Bengio. Parameterizing branch-and-bound search trees to learn branching policies. In *Proceedings of the aaai conference on artificial intelligence*, volume 35, pages 3931–3939, 2021.

# A Theoretical Foundations and Details

## A.1 B&B Dual Price Function: Formal Construction

This section provides the complete formal construction of the B&B dual price function $F_{\mathrm{BB}}$ referenced in Sec 4.1. The piecewise-affine structure and strong duality properties are well-established; we include the full development here for completeness.

Fix a nominal MILP instance $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{\ell}, \boldsymbol{u})$ of Eq. 1. Consider a deterministic LP-based B& B algorithm run to proven optimality. Let $\mathcal{T}$ denote the final set of leaf nodes.

Each node $t$ induces local bounds $(\boldsymbol{\ell}^t, \boldsymbol{u}^t)$ with $\boldsymbol{\ell} \le \boldsymbol{\ell}^t \le \boldsymbol{u}^t \le \boldsymbol{u}$. Define the node LP relaxation

$$z_t(\boldsymbol{b}) := \min_{\boldsymbol{x} \in \mathbb{R}^n} \quad \boldsymbol{c}^\top \boldsymbol{x} \tag{15a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}, \boldsymbol{\ell}^t \le \boldsymbol{x} \le \boldsymbol{u}^t. \tag{15b}$$

For a node whose LP relaxation is feasible and bounded at $\boldsymbol{b}_0$, let $(\boldsymbol{\lambda}^t, \underline{\boldsymbol{\mu}}^t, \overline{\boldsymbol{\mu}}^t)$ be an optimal dual solution of Eq. 15, where $\boldsymbol{\lambda}^t$ is associated with $\boldsymbol{A}\boldsymbol{x} \le \boldsymbol{b}_0$, $\underline{\boldsymbol{\mu}}^t$ with $\boldsymbol{x} \ge \boldsymbol{\ell}^t$, and $\overline{\boldsymbol{\mu}}^t$ with $\boldsymbol{x} \le \boldsymbol{u}^t$. Under the standard LP duality convention for a minimization primal with "$\le$" constraints, the dual variables satisfy

$$\boldsymbol{\lambda}^t \le \boldsymbol{0}, \ \underline{\boldsymbol{\mu}}^t \ge \boldsymbol{0}, \ \overline{\boldsymbol{\mu}}^t \ge \boldsymbol{0}, \tag{16}$$

and

$$\boldsymbol{A}^\top \boldsymbol{\lambda}^t + \underline{\boldsymbol{\mu}}^t - \overline{\boldsymbol{\mu}}^t = \boldsymbol{c} \tag{17}$$

Strong LP duality at the nominal right-hand side (RHS) gives

$$z_t(\boldsymbol{b}_0) = (\boldsymbol{\lambda}^t)^\top \boldsymbol{b}_0 + (\underline{\boldsymbol{\mu}}^t)^\top \boldsymbol{\ell}^t - (\overline{\boldsymbol{\mu}}^t)^\top \boldsymbol{u}^t. \tag{18}$$

If some components of $\boldsymbol{\ell}^t$ or $\boldsymbol{u}^t$ are infinite, the corresponding bound constraints and dual terms are omitted.

Following the construction of the dual price-function [20], define the node-wise affine price form

$$\phi_t(\boldsymbol{d}) := (\boldsymbol{\lambda}^t)^\top \boldsymbol{d} + (\underline{\boldsymbol{\mu}}^t)^\top \boldsymbol{\ell}^t - (\overline{\boldsymbol{\mu}}^t)^\top \boldsymbol{u}^t, \ t \in \mathcal{T}, \tag{19}$$

Since the dual feasibility condition Eq. 17 does not depend on the right-hand side, the same dual vector remains feasible for the node LP with RHS $\boldsymbol{d}$. Therefore,

$$\phi_t(\boldsymbol{d}) \le z_t(\boldsymbol{d}) \tag{20}$$

Let $\mathcal{T}$ denote the dual-certified terminal collection of the recorded B&B tree. For notational simplicity, terminal nodes pruned by infeasibility are either excluded from $\mathcal{T}$ or understood to be handled by standard Farkas certificates. Define the induced B&B dual price function

$$F_{\mathrm{BB}}(\boldsymbol{d}) := \min_{t \in \mathcal{T}} \phi_t(\boldsymbol{d}). \tag{21}$$

**Theorem A.1** (Piecewise-affine B&B dual price function)**.** *The function $F_{\mathrm{BB}}$ defined in Eq. 21 is a pointwise minimum of finitely many affine functions, and is therefore piecewise affine and concave. Moreover, if $\mathcal{T}$ is a valid terminal cover of the B&B partition equipped with valid dual certificates, then*

$$F_{\mathrm{BB}}(\boldsymbol{d}) \le z(\boldsymbol{d}) \tag{22}$$

*for every right-hand side $\mathbf{d}$ for which the recorded terminal cover remains valid.*

*Proof.* The piecewise-affine property follows directly from the definition

$$F_{\mathrm{BB}}(\boldsymbol{d}) = \min_{t \in \mathcal{T}} \phi_t(\boldsymbol{d}), \tag{23}$$

because each $\phi_t(\boldsymbol{d})$ is affine in $\boldsymbol{d}$. A pointwise minimum of finitely many affine functions is piecewise affine and concave.

It remains to prove the weak-dual lower-bound property. Fix any right-hand side $\boldsymbol{d}$ for which the recorded terminal cover remains valid. If $\mathcal{X}(\boldsymbol{d})$ is empty, then $z(\boldsymbol{d}) = +\infty$ under the minimization convention, and the claim is trivial. Otherwise, take any feasible integer solution $\boldsymbol{x} \in \mathcal{X}(\boldsymbol{d})$. Since the terminal nodes of the recorded B&B tree form a valid cover of the branching partition, there exists at least one terminal node $t \in \mathcal{T}$ such that $\boldsymbol{x}$ satisfies the local bounds $(\boldsymbol{\ell}^t, \boldsymbol{u}^t)$ of node $t$. Hence $\boldsymbol{x}$ is feasible for the node LP relaxation with right-hand side $\boldsymbol{d}$, and therefore

$$z_t(\boldsymbol{d}) \le \boldsymbol{c}^\top \boldsymbol{x}. \tag{24}$$

By dual feasibility of $(\boldsymbol{\lambda}^t, \underline{\boldsymbol{\mu}}^t, \overline{\boldsymbol{\mu}}^t)$ for the node LP with RHS $\boldsymbol{d}$, we have

$$\phi_t(\boldsymbol{d}) \le z_t(\boldsymbol{d}). \tag{25}$$

Combining the two inequalities gives

$$\phi_t(\boldsymbol{d}) \le \boldsymbol{c}^\top \boldsymbol{x}. \tag{26}$$

Moreover, by definition of $F_{\mathrm{BB}}$,

$$F_{\mathrm{BB}}(\boldsymbol{d}) = \min_{s \in \mathcal{T}} \phi_s(\boldsymbol{d}) \le \phi_t(\boldsymbol{d}). \tag{27}$$

Thus

$$F_{\mathrm{BB}}(\boldsymbol{d}) \le \boldsymbol{c}^\top \boldsymbol{x}. \tag{28}$$

Taking the minimum over all feasible integer solutions $\boldsymbol{x} \in \mathcal{X}(\boldsymbol{d})$ yields

$$F_{\mathrm{BB}}(\boldsymbol{d}) \le z(\boldsymbol{d}). \tag{29}$$

This proves the theorem. □

The function $F_{\mathrm{BB}}$ provides a recorded dual surrogate for measuring how much a perturbation changes the B&B-induced lower-bound structure. In particular, for an RHS perturbation $\Delta\boldsymbol{b}$, the change of the affine price form at node $t$ is

$$\phi_t(\boldsymbol{b}_0 + \Delta\boldsymbol{b}) - \phi_t(\boldsymbol{b}_0) = (\boldsymbol{\lambda}^t)^\top \Delta\boldsymbol{b}. \tag{30}$$

### A.2 Proof of Theorem 4.1

*Proof.* Fix a node $t \in \mathcal{T}_{\le K}$. We write the node LP in standard form after introducing slack variables for the original constraints:

$$\boldsymbol{A}\boldsymbol{x} + \boldsymbol{\sigma} = \boldsymbol{b},\ \boldsymbol{\sigma} \ge \boldsymbol{0}. \tag{31}$$

Let $\boldsymbol{y}_B^t(\boldsymbol{b}_0)$ denote the basic solution associated with the selected basis $B^t$ at the nominal RHS $\boldsymbol{b}_0$. Then

$$\boldsymbol{y}_B^t(\boldsymbol{b}_0) = (B^t)^{-1} \widetilde{\boldsymbol{b}}^{\,t}(\boldsymbol{b}_0), \tag{32}$$

where $\widetilde{\boldsymbol{b}}^{\,t}(\boldsymbol{b}_0)$ is the standard-form right-hand side of the node LP.

For any $i \in \mathcal{I}_{K,\tau}$, the slack variable $\sigma_i$ is basic in $B^t$. The column of $\sigma_i$ in the standard-form constraint matrix is the unit vector corresponding to the $i$-th original constraint row. Therefore,

$$(B^t)^{-1} \boldsymbol{e}_i = \boldsymbol{e}_{r_t(i)}, \tag{33}$$

where $r_t(i)$ denotes the position of the basic variable $\sigma_i$ inside the basis $B^t$.

Since $\Delta\boldsymbol{b} \in \mathcal{Q}_{K,\tau,\alpha}$ is supported only on $\mathcal{I}_{K,\tau}$, the perturbed basic solution satisfies

$$\boldsymbol{y}_B^t(\boldsymbol{b}_0 + \Delta\boldsymbol{b}) = \boldsymbol{y}_B^t(\boldsymbol{b}_0) + \sum_{i \in \mathcal{I}_{K,\tau}} \Delta b_i \boldsymbol{e}_{r_t(i)}. \tag{34}$$

Thus the perturbation changes only the corresponding basic slack variables. In particular, the original decision-variable component of the selected basic solution is unchanged:

$$\boldsymbol{x}^t(\boldsymbol{b}_0 + \Delta\boldsymbol{b}) = \boldsymbol{x}^t(\boldsymbol{b}_0). \tag{35}$$

We next prove primal feasibility of the same basis. For each $i \in \mathcal{I}_{K,\tau}$,

$$s_i^t + \Delta b_i \geq s_i^t - \alpha \underline{s}_i \geq \underline{s}_i - \alpha \underline{s}_i = (1-\alpha)\underline{s}_i > 0, \tag{36}$$

where we used $0 < \alpha < 1$ and $\underline{s}_i \geq \tau > 0$. All other basic variables are unchanged, and they were feasible at $\boldsymbol{b}_0$. Therefore, the same basis $B^t$ remains primal feasible for the perturbed node LP.

Finally, RHS perturbations do not change the reduced costs of a fixed basis, because the objective vector and the constraint matrix are unchanged. Since $B^t$ was optimal at $\boldsymbol{b}_0$, the same basis remains optimal at $\boldsymbol{b}_0 + \Delta\boldsymbol{b}$. The node LP objective contains no cost for slack variables, and the decision-variable component is unchanged. Hence

$$z_t(\boldsymbol{b}_0 + \Delta\boldsymbol{b}) = z_t(\boldsymbol{b}_0). \tag{37}$$

This proves the theorem. □

### A.3 Prefix-stability implication

The inactive-basic certificate in Theorem 4.1 gives node-wise LP invariance on the recorded prefix. To translate this node-wise invariance into a B&B-prefix statement, we use the following deterministic solver abstraction.

**Definition A.2** (Certified $K$-prefix identity)**.** Given two B&B runs generated by the same deterministic algorithm at $\boldsymbol{b}_0$ and $\boldsymbol{b}_0 + \Delta\boldsymbol{b}$, we say that they are certified $K$-prefix identical if, for all nodes in the recorded prefix up to depth $K$, the two runs have the same local bounds, the same selected LP solution and LP value, the same branch/prune/integer-feasible status, and, for branching nodes, the same branching variable and branching value.

**Assumption A.3** (Deterministic certified B&B prefix)**.** Inside the recorded prefix $\mathcal{T}_{\leq K}$, the analyzed B&B process satisfies the following conditions.

First, the LP solver uses deterministic basis selection: if the previously selected basis remains primal feasible and optimal, the same basis and the associated basic solution are returned.

Second, node selection, branching, pruning, and tie-breaking are deterministic functions of the local bounds, selected LP solution, selected LP value, and incumbent value.

Third, no cuts, presolve modifications, or external primal heuristics alter the effective node LP inside the certified prefix.

Fourth, incumbent solutions used for pruning inside the prefix are either generated by certified integer-feasible LP solutions inside the prefix or are separately verified to remain feasible with the same objective value after perturbation.

Finally, if infeasible-node pruning occurs inside the certified prefix, the corresponding infeasibility certificate is assumed to remain valid under the perturbation.

**Corollary A.4** (Computable RHS region implying certified prefix identity)**.** *Under Assumption A.3, if*

$$\Delta\boldsymbol{b} \in \mathcal{Q}_{K,\tau,\alpha}, \tag{38}$$

*then the two B&B runs at $\boldsymbol{b}_0$ and $\boldsymbol{b}_0 + \Delta\boldsymbol{b}$ are certified $K$-prefix identical.*

*Proof.* We prove the claim by induction over the deterministic node-processing order inside the recorded prefix.

At the root node, Theorem 4.1 implies that the selected LP solution and LP value are unchanged after perturbation. Therefore, the root has the same integrality status. If it is pruned by bound, the pruning comparison is unchanged because the node LP value and the certified incumbent value are unchanged. If it branches, the deterministic branching rule receives the same selected LP solution and therefore chooses the same branching variable and branching value.

Now assume that all processed nodes up to a certain point in the certified prefix coincide between the two runs. Then the same local bounds have been created for the next corresponding node. By Theorem 4.1, this node has the same selected LP solution and the same LP value under $\boldsymbol{b}_0 + \Delta\boldsymbol{b}$ as under $\boldsymbol{b}_0$. Hence its branch/prune/integer-feasible status is unchanged. If the node branches, the same branching variable and branching value are selected. If the node is pruned by infeasibility,

Assumption A.3 ensures that the infeasibility certificate remains valid. If it is pruned by bound, the pruning comparison is unchanged by the certified invariance of the LP value and incumbent value.

Because node selection and all tie-breaking rules are deterministic, the next selected node is also the same. The induction therefore shows that all nodes and decisions in the recorded prefix up to depth $K$ coincide. Thus the two runs are certified $K$-prefix identical. □

### A.4 Proof of Theorem 4.2

*Proof.* Fix any feasible solution $\boldsymbol{x} \in \mathcal{X}_I$. Since $\boldsymbol{x}^\star$ is optimal for the nominal objective $\boldsymbol{c}$, we have

$$\boldsymbol{c}^\top(\boldsymbol{x} - \boldsymbol{x}^\star) \geq 0. \tag{39}$$

We next show that

$$\Delta\boldsymbol{c}^\top(\boldsymbol{x} - \boldsymbol{x}^\star) \geq 0. \tag{40}$$

For $j \in J_L$, we have $x_j^\star = \ell_j$ and $x_j \geq \ell_j$, so

$$x_j - x_j^\star \geq 0. \tag{41}$$

Since $\Delta c_j \geq 0$ for $j \in J_L$, it follows that

$$\Delta c_j(x_j - x_j^\star) \geq 0. \tag{42}$$

For $j \in J_U$, we have $x_j^\star = u_j$ and $x_j \leq u_j$, so

$$x_j - x_j^\star \leq 0. \tag{43}$$

Since $\Delta c_j \leq 0$ for $j \in J_U$, it follows that

$$\Delta c_j(x_j - x_j^\star) \geq 0. \tag{44}$$

For $j \in J_F$, the definition of $\mathcal{C}(\boldsymbol{x}^\star)$ gives $\Delta c_j = 0$, so

$$\Delta c_j(x_j - x_j^\star) = 0. \tag{45}$$

For $j \in J_{\text{fix}}$, every feasible solution satisfies $x_j = x_j^\star$, and hence

$$\Delta c_j(x_j - x_j^\star) = 0. \tag{46}$$

Summing over all coordinates gives

$$\Delta\boldsymbol{c}^\top(\boldsymbol{x} - \boldsymbol{x}^\star) \geq 0. \tag{47}$$

Therefore,

$$(\boldsymbol{c} + \Delta\boldsymbol{c})^\top(\boldsymbol{x} - \boldsymbol{x}^\star) = \boldsymbol{c}^\top(\boldsymbol{x} - \boldsymbol{x}^\star) + \Delta\boldsymbol{c}^\top(\boldsymbol{x} - \boldsymbol{x}^\star) \geq 0. \tag{48}$$

Since this holds for every $\boldsymbol{x} \in \mathcal{X}_I$, the solution $\boldsymbol{x}^\star$ remains optimal for the perturbed objective $\boldsymbol{c} + \Delta\boldsymbol{c}$. Evaluating the perturbed objective at $\boldsymbol{x}^\star$ gives

$$z(\boldsymbol{c} + \Delta\boldsymbol{c}, \boldsymbol{b}_0) = (\boldsymbol{c} + \Delta\boldsymbol{c})^\top\boldsymbol{x}^\star = z(\boldsymbol{c}, \boldsymbol{b}_0) + \Delta\boldsymbol{c}^\top\boldsymbol{x}^\star. \tag{49}$$

□

### A.5 Objective drift of random perturbations

Random sign perturbations are used as non-certified counterparts to the objective-preserving perturbations. Let $S \subseteq \{1, \ldots, n\}$ be a perturbation support, and draw

$$\Delta c_j = \rho\varepsilon_j, \ \forall j \in S, \ \Delta c_j = 0, \ \forall j \notin S, \tag{50}$$

where $\varepsilon_j$ are independent Rademacher random variables satisfying

$$\mathbb{P}(\varepsilon_j = 1) = \mathbb{P}(\varepsilon_j = -1) = \frac{1}{2}. \tag{51}$$

Then

$$\Delta \boldsymbol{c}^\top \boldsymbol{x}^\star = \rho \sum_{j \in S} \varepsilon_j x_j^\star. \tag{52}$$

By Khintchine's inequality, there exists a universal constant $\kappa > 0$ such that

$$\mathbb{E}\left[\left|\sum_{j \in S} \varepsilon_j x_j^\star\right|\right] \geq \kappa \left(\sum_{j \in S} (x_j^\star)^2\right)^{1/2}. \tag{53}$$

Therefore,

$$\mathbb{E}\left[\left|\Delta \boldsymbol{c}^\top \boldsymbol{x}^\star\right|\right] \geq \kappa \rho \|\boldsymbol{x}_S^\star\|_2, \tag{54}$$

where $\boldsymbol{x}_S^\star$ denotes the restriction of $\boldsymbol{x}^\star$ to the support $S$.

Thus, whenever $\boldsymbol{x}_S^\star \neq \mathbf{0}$, a random sign perturbation induces nonzero expected drift at the nominal optimal solution. In contrast, the zero-support perturbations satisfy

$$\Delta \boldsymbol{c}^\top \boldsymbol{x}^\star = 0, \tag{55}$$

and hence preserve the optimal objective value. If $\boldsymbol{x}^\star$ also remains optimal after the random perturbation, then

$$z(\boldsymbol{c} + \Delta \boldsymbol{c}, \boldsymbol{b}_0) - z(\boldsymbol{c}, \boldsymbol{b}_0) = \Delta \boldsymbol{c}^\top \boldsymbol{x}^\star, \tag{56}$$

so the same lower bound quantifies the expected absolute optimal-value change.

# B L2B Experimental Details

All experiments are done on our workstation with AMD 3970X, RTX 3090, and 128GB memory and AMD Ryzen Threadripper 3970X 32-Core Processor.

## B.1 L2B Benchmarks

We conducted our L2B experiments on four well-established NP-hard MILP benchmarks from the literature [14]: Set Covering (SC), Combinatorial Auction (CA), Capacitated Facility Location (CFL), and Maximum Independent Set (MIS). The data generation procedures strictly followed the established protocol in [14]. Each benchmark comprises instances across three difficulty levels: *Easy*, *Medium*, and *Hard*. For the *Easy* level, we generated 10,000 instances for training, 2,000 for validation, and 20 for testing. For both *Medium* and *Hard* levels, 20 test instances were generated respectively. Table 4 summarizes the statistical characteristics of all generated instances.

Expert policy labels were generated following [14] using SCIP's *vanillafullstrong* branching rule. During this label generation process, which coincides with problem solving, we recorded complete node information necessary for both calculating the MILP dual function and generating positive pairs through our MIRAGE-C augmentation strategy. This integrated approach ensures that our method leverages existing solving processes without introducing substantial additional computational overhead.

Table 4: Statistical information of L2B benchmarks.

| **Difficulty Level** | | **SC** | **CA** | **CFL** | **MIS** |
|---|---|---|---|---|---|
| *Easy* | Constraint Number | 500 | 193 | 10201 | 1953 |
| | Variable Number | 1000 | 500 | 10100 | 500 |
| *Medium* | Constraint Number | 1000 | 384 | 20301 | 3946 |
| | Variable Number | 1000 | 1000 | 20100 | 1000 |
| *Hard* | Constraint Number | 2000 | 577 | 40501 | 5940 |
| | Variable Number | 1000 | 1500 | 40100 | 1500 |

Table 5: Bipartite graph features utilized for L2B experiments.

| Type | Feature Name | Attribute | Description |
|---|---|---|---|
| Constraint Feature | obj_cos_sim | Local | Cosine similarity between the constraint coefficients and the objective function vector. |
| | bias | Global | The right-hand side (bias) value, normalized by the constraint coefficients. |
| | is_tight | Local | A binary indicator for whether the constraint is tight (binding) in the LP relaxation solution. |
| | dualsol_val | Local | The dual solution value for the constraint, normalized. |
| | age | Local | The age of the LP solution, normalized by the total number of LPs solved. |
| Edge Feature | coef | Global | The coefficient of a variable in a constraint, normalized per constraint. |
| Variable Feature | type | Global | A one-hot encoding of the variable type (binary, integer, implicit integer, continuous). |
| | coef | Global | The objective function coefficient for the variable, normalized. |
| | has_lb | Global | Binary indicators for the presence of a lower bound. |
| | has_ub | Global | Binary indicators for the presence of an upper bound. |
| | sol_is_at_lb | Local | Binary indicators for whether the current LP solution value is at the variable's lower bound. |
| | sol_is_at_ub | Local | Binary indicators for whether the current LP solution value is at the variable's upper bound. |
| | sol_frac | Local | The fractionality of the variable's current LP solution value. |
| | basis_status | Local | A one-hot encoding of the variable's status (lower, basic, upper, zero) in the simplex basis. |
| | reduced_cost | Local | The reduced cost of the variable, normalized. |
| | age | Local | The age of the variable's current value, normalized. |
| | sol_val | Local | The current LP solution value for the variable. |
| | inc_val | Local | The variable's value in the best incumbent (feasible) solution. |
| | avg_inc_val | Local | The variable's average value across all incumbent solutions. |

### B.2 Network Architecture

All machine learning methods in our L2B experiments employ an identical Graph Convolutional Neural Network (GCNN) architecture following the implementation of [14].

Since the vanilla GCNN architecture does not naturally produce instance-level representations, we extend it with additional pooling and projection layers. After obtaining the final constraint and variable embeddings through message passing, we apply MaxPooling to constraint features and AveragePooling to variable features to generate 1D global embeddings for each modality. This asymmetric pooling strategy is intentional: MaxPooling on constraint features preserves the most salient information, which is particularly relevant for our MIRAGE-C augmentation strategy that involves constraint perturbations, while AveragePooling on variable features captures comprehensive characteristic patterns. We intentionally guide the model to capture the most significant constraint features, which are invariant or robust in data augmentation instead of smoothing the features by averaging. The resulting global constraint and variable embeddings are then concatenated and processed through a two-layer MLP to produce the final instance-level representation.

Regarding baseline comparisons, we note that the official implementation of CAMBranch is not publicly available. Our implementation of this baseline therefore follows the methodological descriptions in [27] as closely as possible.

### B.3 Feature Set

We employ the same bipartite graph feature set as [14] for MILP representation. However, not all features in this set are available during our instance-level contrastive pre-training phase, as certain features can only be computed during the actual solving process. As detailed in Table 5, we categorize features into two types based on their scope and availability: global features and local features. Global features, which characterize the static properties of the MILP instance, are available for contrastive learning. Local features, which depend on dynamic solver state information, are unavailable at this stage. Consequently, during contrastive pre-training, we initialize all local features to zero and utilize only global features. In the subsequent imitation learning phase, where solver state information becomes available, we employ the complete feature set including both global and local features.

### B.4 Hyperparameters

In our experimental setup, the hidden dimension for both constraint and variable embeddings was set to 64. The temperature $\tau_{\text{temp}}$ was set to 0.5. For the contrastive pre-training phase, we found that

employing an appropriate learning rate was crucial for convergence. Empirically, the optimal learning rate for this phase was significantly lower than that used in subsequent imitation learning, suggesting that the instance representations learned during pre-training are particularly sensitive to optimization dynamics. Specifically, we used a learning rate of $2 \times 10^{-5}$ for the SC, CA, and MIS benchmarks, and $1 \times 10^{-5}$ for CFL during contrastive pre-training. For the imitation learning phase, the learning rate was set to $1 \times 10^{-3}$ across all benchmarks. This two-order-of-magnitude difference in learning rates between the two phases underscores the delicate nature of representation learning.

# C PaS Experimental Details

All experiments are done on our workstation with AMD 3970X, RTX 3090, and 128GB memory and AMD Ryzen Threadripper 3970X 32-Core Processor.

## C.1 PaS Benchmarks

Our PaS experiments were conducted on four MILP benchmarks established in [21], comprising two synthetic problems—Combinatorial Auction (CA) and Maximum Independent Set (MIS)—alongside two real-world challenges from the NeurIPS ML4CO 2021 competition [13]: Item Placement (IP) and Workload Appointment (WA). Synthetic instances were randomly generated using the code in [14]. Notably, the MIS instances from [21] were found to be insufficiently challenging to adequately demonstrate the capabilities of ML methods; we therefore increased the problem scale to create more difficult instances. For the IP and WA benchmarks, we used the publicly available instances from the competition. Following [21], each benchmark consists of 400 instances partitioned into 240 for training, 60 for validation, and 100 for testing. Table 6 summarizes the statistical characteristics of these benchmarks.

Table 6: Statistical information of PaS benchmarks.

| | **IP** | **WA** | **CA** | **MIS** |
|---|---|---|---|---|
| Constraint Number | 195 | 64305 | 2555 | 8480 |
| Variable Number | 1083 | 61000 | 1500 | 2000 |

## C.2 Network Architecture

Similar to our L2B experiments, all machine learning methods in our PaS experiments employ identical model architectures based on the implementation of [21] to ensure fair comparison.

To generate instance-level representations, we extend the base architecture with additional pooling and projection layers. After obtaining the final constraint and variable embeddings through message passing, we apply AveragePooling to constraint features and MaxPooling to variable features to produce global embeddings for each modality. This pooling strategy differs from our L2B implementation and is specifically designed to align with our MIRAGE-O augmentation approach: since this strategy perturbs objective coefficients (variable features), MaxPooling on variables helps preserve the most critical variable information, while AveragePooling on constraints captures general constraint patterns. The resulting global embeddings are then concatenated and processed through a two-layer MLP to produce the final instance-level representation.

We note that the official implementation of ConPaS is not publicly available. Our implementation of this baseline accordingly follows the methodological descriptions in [23] as closely as possible.

## C.3 Feature Set

We employ the same bipartite graph feature set as [21], as shown in Table 8. All of the features are global features and are available during our contrastive pre-training phase.

## C.4 Hyperparameters

In our experimental configuration, the hidden dimension for all embeddings was set to 64. The temperature $\tau_{\text{temp}}$ was set to 0.5. During the contrastive pre-training phase, we employed learning

Table 7: Hyperparameters $(k_0, k_1, \Delta)$ used for PaS experiments.

| Model | IP | WA | CA | MIS |
|---|---|---|---|---|
| PS+Gurobi | (400, 5, 10) | (0, 500, 10) | (600, 0, 1) | (400, 400, 20) |
| ConPaS+Gurobi | (400, 5, 2) | (0, 500, 10) | (1000, 0, 1) | (500, 400, 25) |
| MIRAGE-PS+Gurobi | (400, 5, 10) | (0, 500, 10) | (600, 0, 1) | (400, 400, 20) |
| PS+SCIP | (400, 5, 1) | (0, 500, 5) | (400, 0, 10) | (400, 400, 15) |
| ConPaS+SCIP | (400, 5, 2) | (0, 500, 20) | (1000, 0, 1) | (500, 400, 25) |
| MIRAGE-PS+SCIP | (400, 5, 1) | (0, 500, 5) | (400, 0, 10) | (400, 400, 15) |

Table 8: Bipartite graph features utilized for PaS experiments.

| Type | Feature Name | Description |
|---|---|---|
| Constraint Feature | c_coeff | Mean value of all coefficients within the constraint. |
| | Nc_coeff | Degree of the constraint node within the bipartite graph. |
| | rhs | Right-hand-side value of the constraint. |
| | sense | Type of the constraint (e.g., inequality or equality). |
| Edge Feature | coeff | Coefficient value of the variable in the specific constraint. |
| Variable Feature | obj | Normalized objective function coefficient for the variable. |
| | v_coeff | Mean coefficient value of the variable across all constraints. |
| | Nv_coeff | Degree of the variable node within the bipartite graph. |
| | max_coeff | Maximum coefficient value associated with the variable. |
| | min_coeff | Minimum coefficient value associated with the variable. |
| | int | Binary indicator specifying whether the variable is an integer type. |
| | pos_emb | A 12-dimensional binary encoding representing the variable's positional order among all variables. |

Table 9: Cost profile of MIRAGE.

| **Stage** | **Wall-clock** | **Peak GPU mem** | **Peak CPU mem** |
|---|---|---|---|
| Dual recording (piggybacks on routine label generation) | +1.8% of labeling time | 0 (CPU-side) | 0.9 GB |
| Positive-sample generation | 0.42 s/inst. (MIRAGE-C); 1.35 s/inst. (MIRAGE-O) | 0 (CPU-side) | negligible |
| Contrastive pre-training | 12min (L2B); 18 min (PaS) | 3.2 GB | 6.4 GB |
| Downstream training | identical to baseline | identical | identical |
| **Additional test-time cost** | **None** | **None** | **None** |

rates of $2 \times 10^{-5}$ for the CA and MIS benchmarks, and $1 \times 10^{-5}$ for the IP and WA benchmarks. For the subsequent training procedure, a consistent learning rate of $3 \times 10^{-3}$ was applied across all benchmarks. Moreover, we summarize the key configuration parameters—specifically, the variable fixing-parameter $(k_0, k_1)$ and the neighborhood search parameter $\Delta$—in Table 7.

# D Computational and Memory Overhead

Table 9 provides a complete cost profile of MIRAGE on the MIS benchmark (RTX 3090, 128 GB workstation).

MIRAGE introduces offline training-time overhead for solver-information recording, sample generation, and contrastive pre-training, while leaving the deployed model architecture unchanged. At test time, MIRAGE introduces no additional parameters, operations, or model invocations relative to the downstream backbone.

# E Additional Ablation Studies

## E.1 Per-Setting Ablation Breakdown

Table 10 reports the per-setting comparison between A3 (MIRAGE-Full) and A4 (RandPos-CL). A3 is numerically better in all evaluated settings.

Table 10: A3 vs A4 per setting. W/T/L, referring to Win/Tie/Lose, is computed over 20 test instances for L2B, and 100 for PaS.

| Setting | A4 | A3 | W/T/L |
|---|---|---|---|
| SC-E/M/H | 8.81 / 55.66 / 1901.43 | **8.57 / 55.47 / 1876.78** | 15/1/4; 11/2/7; 12/1/7 |
| CA-E/M/H | 1.98 / 11.61 / 125.60 | **1.72 / 11.27 / 106.91** | 16/1/3; 12/2/6; 16/0/4 |
| CFL-E/M/H | 28.72 / 139.14 / 719.87 | **26.90 / 136.89 / 627.84** | 14/1/5; 15/2/3; 15/1/4 |
| MIS-E/M/H | 7.51 / 194.90 / 1355.06 | **6.88 / 122.63 / 732.29** | 15/1/4; 18/0/2; 19/0/1 |
| IP/WA/CA/MIS-Gurobi | 0.51 / 0.70 / 259.39 / 0.01 | **0.37 / 0.52 / 186.53 / 0.00** | 69/18/13; 67/20/13; 73/14/13; 12/88/0 |
| IP/WA/CA/MIS-SCIP | 2.28 / 1.25 / 1000.27 / 0.05 | **1.82 / 0.88 / 819.20 / 0.00** | 66/17/17; 72/14/14; 64/19/17; 16/84/0 |

Table 11: Parameter sensitivity results. SC/CA report time; IP/WA report SCIP $\text{gap}_{\text{abs}}$.

| Parameter | Tested values | SC Time | CA Time | IP $\text{gap}_{\text{abs}}$ | WA $\text{gap}_{\text{abs}}$ |
|---|---|---|---|---|---|
| $K_{\max}$ | 2 / 5 / 10 / 15 / 20 | 8.76 / 8.64 / 8.57 /8.52/ 8.55 | 1.93 / 1.76 /1.72 / 1.73 /1.72 | — | — |
| $\tau$ | $10^{-5}/10^{-4}/10^{-3}/10^{-2}/10^{-1}$ | 8.59 /8.56/ 8.57 / 8.59 / 8.61 | 1.76 / 1.73 /1.72/1.72/ 1.75 | — | — |
| $\alpha$ | 0.1 / 0.3 / 0.5 / 0.7 / 0.9 | 8.66 / 8.61 / 8.57 /8.53/ 8.59 | 1.80 / 1.77 / 1.72 /1.70/ 1.71 | — | — |
| $\eta_c$ | 2% / 5% / 8% / 15% / 30% | — | — | 1.94 / 1.86 / 1.82 /1.79/ 1.90 | 0.97 / 0.91 /0.88/ 0.89 / 0.96 |

### E.2 Mechanism checks under standard SCIP

We rerun perturbed instances with standard SCIP to measure the intended invariances:

- **MIRAGE-C (on SC-Easy)**: Certified positives preserve the K-prefix in 96.9% of cases, compared to 41.7% for feasible support-cardinality-/norm-matched random and 28.9% for unfiltered random.
- **MIRAGE-O (on IP)**: Certified positives maintain $\boldsymbol{x}^\star$ optimal in 100% of cases with exact value preservation, compared to 58.6%/7.8% for support-cardinality-/norm-matched random signs and 34.2%/1.9% for full-support random signs.

These results confirm that certificate-constrained positives substantially better preserve the intended solver-level invariances than matched non-certified positives.

### E.3 Hyperparameter Sensitivity

**Certificate parameters.**

- **MIRAGE-C**: For anchor instance $i$, we use $K_i = \min(K_{\max}, D_i)$, where $D_i$ is the depth of its recorded B&B tree and $K_{\max} = 10$. We use the inactive-basic threshold $\tau_s = 10^{-3}$ and perturbation ratio $\alpha = 0.5$ for L2B benchmarks.
- **MIRAGE-O**: Positives are sampled on the value-invariant $J_0$ support and scaled to a target relative magnitude $\eta_c := \|\Delta c\|/\|c\| = 8\%$; random-sign negatives are norm-matched. Equivalently, the raw scale $\rho$ is calibrated per instance to realize $\eta_c$.

Table 11 reports one-factor sweeps over MIRAGE-C parameters on SC/CA (Easy) and MIRAGE-O parameter on IP/WA (SCIP). The results indicate a reasonably stable region around the submitted defaults in the evaluated benchmarks. Relative to the defaults, the maximum degradation is 0.5%/2.3% on SC/CA across the $\tau_s$ sweep and 1.1%/4.7% across the $\alpha$ sweep. For $K_{\max} \geq 5$, CA remains within *.3% of the default, whereas $K_{\max} = 2$ is too shallow in our setting. For MIRAGE-O, $\eta_c \in [5\%, 15\%]$ remains within 2.2% on IP and 3.4% on WA of the default.

## F Broader Impacts

This work lies at the intersection of machine learning and combinatorial optimization. Its primary potential for positive societal impact stems from accelerating the solution of MILPs, which are widely used in resource allocation, logistics, scheduling, and infrastructure planning. By enabling faster and more scalable MILP solving, MIRAGE could contribute to efficiency gains in domains such as supply chain and energy systems.

However, we also acknowledge potential negative or dual-use risks. The same technology that improves benign applications could, in principle, be used to optimize malicious activities. Moreover, automated MILP solvers enhanced by ML may be deployed in settings without adequate human oversight, potentially amplifying systemic biases encoded in the problem formulation.

## 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 abstract and introduction clearly state the proposed MIRAGE framework, its duality-inspired augmentation strategies (constraint and objective perturbations), theoretical insights on preserving solving properties, and empirical improvements on learning-to-branch and predict-and-search tasks. These claims are fully supported by the theoretical derivations in Sec. 4 and experimental results in Sec. 5.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Discussed in Appendix D.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: The paper contains three theorems (Theorems A.1, 4.1, 4.2), each stated with its full assumptions in Sec. 4 and proved in Appendix A.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The paper discloses extensive details necessary for reproducibility: dataset generation protocols, data splits, solver versions, network architectures, feature sets, hyperparameters, contrastive loss formulation, and the procedure for generating positive/negative samples based on dual functions.

Guidelines:

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

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [No]

   Justification: Due to newly identified restrictions, we are unable to release the full codebase. To bridge the gap, we present detailed configurations in Appendix. All experimental data, including the customized MILP benchmarks, are generated using well-documented public protocols as detailed in Experiment section and Appendix.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The paper provides comprehensive training and test details for L2B (Appendices B.1–B.4) and PaS (Appendices C.1–C.4).

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: The paper includes using five different seeds and 100 solving attempts per configuration.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: The needed computing resources are presented in the Appendix.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics, and are sure that the research conforms to it.

   Guidelines:

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

10. **Broader impacts**

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

Answer: [Yes]

Justification: Discussed in the Appendix.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

Answer: [N/A]

Justification: The paper poses no such risks.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: The assets (baselines, datasets) are properly cited.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: As described in the paper.

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

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

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

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

    Answer: [N/A]

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

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [N/A]

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

    Guidelines:

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