HyCO: A Hybrid Neural Solver for Combinatorial Optimization

Yuheng Li[*]
Department of Data Science
College of William & Mary
yli95@wm.edu

Di Yang[*]
Department of Data Science
College of William & Mary
dyang06@wm.edu

Haipeng Chen[†]
Department of Data Science
College of William & Mary
hchen23@wm.edu

Yanhai Xiong[†]
Department of Data Science
College of William & Mary
yxiong05@wm.edu

Abstract

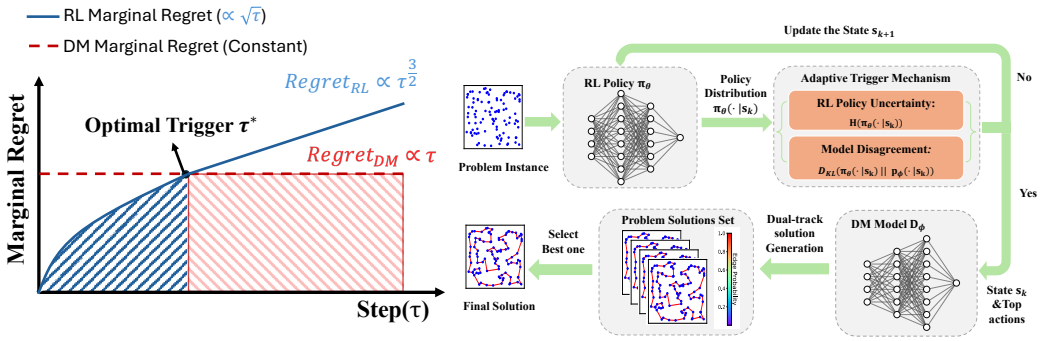
Sequential reinforcement learning (RL) solvers and global diffusion model (DM) solvers for neural combinatorial optimization exhibit complementary failure modes under an optimization-regret view. The former enjoys small marginal regret in the early construction stage, but suffers from horizon-wise compounding errors with super-linear regret growth; the latter avoids horizon compounding but incurs linear or sublinear regret w.r.t. the dimension of the remaining unsolved subspace. We propose Hybrid Neural Solver for Combinatorial Optimization (**HyCO**), a hybrid inference algorithm that constructs a solution prefix with an RL solver and adaptively switches to a conditional DM to complete the remaining decisions. To characterize why such hybridization helps, when to trigger the handover, and how to realize it in practice, we first develop a unified error-scaling theoretical framework and prove that, under explicit error-scaling assumptions, i) the hybrid structure achieves strictly lower expected regret than either backbone alone, and ii) there exists a unique optimal trigger step that minimizes the hybrid regret. We then design a lightweight adaptive trigger that combines policy entropy and RL–DM disagreement to detect trajectory-level signals of the regime shift as a practical proxy, since the optimal trigger step is defined at the expected-regret level and is not directly computable on individual trajectories. Experimental results on diverse benchmarks demonstrate that HyCO achieves consistent improvements over both backbones and support the empirical effectiveness of adaptive triggering. Our code and checkpoints are available at `https://github.com/wmd3i/HyCO`.

1 Introduction

Combinatorial optimization (CO) problems, such as the Traveling Salesman Problem (TSP), are fundamental to operations research but notoriously difficult to solve at scale due to their NP-hard nature [1, 2]. While traditional exact solvers like Concorde [3] provide optimality, their exponential complexity limits scalability. This has spurred the design of neural solvers that learn heuristics directly from data. Among recent neural approaches, two representative solver classes have emerged: sequential construction solvers based on reinforcement learning (RL) [4–6], and global generative solvers based on diffusion models (DMs) [7, 8].

[*]Equal contribution.
[†]Corresponding authors.

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



(a) Regret Dynamics (Motivation)

(b) HyCO Inference Framework

Figure 1: **Motivation and Method. (a)** Schematic illustration of the unified error dynamics. The marginal regret of RL increases as the trajectory lengthens, while that of a conditional DM is constant as the solution prefix grows. Their intersection defines the optimal trigger step τ^*. **(b)** The HyCO framework, which constructs a prefix with a RL solver, detects the regime shift via a lightweight adaptive trigger, and hands over to a conditional DM for dual-track completion.

Both classes aim to produce high-quality solutions, but they rely on fundamentally different inference mechanisms and exhibit **complementary error trends** under an optimization-regret view:

RL-based sequential solvers typically construct solutions in an autoregressive manner. At early decision steps, the marginal regret is typically small as the solver operates within the high-density regions of the ground-truth state distribution. However, such solvers are inherently vulnerable to **horizon-wise compounding errors** [9, 10]: as the decision horizon extends, even small local deviations steer the state trajectory away from the training state distribution, causing decision errors to propagate and accumulate super-linearly w.r.t. the horizon [11–13].

In contrast, DM-based global solvers exhibit a different error profile. Unlike RL solvers, which compound errors over sequential decision steps, DMs incur a solution-space-governed regret that can scale **linearly** (or sub-linearly) with the effective dimension of the solution space [14]. While generating a full high-dimensional solution from scratch incurs high regret due to the vast search space, the marginal regret of a conditional DM remains constant or may even decrease as the solution prefix grows. As the constraints from the prefix tighten, the remaining subspace shrinks, simplifying the generative task. This inverse relationship implies a structural crossing point where the optimal strategy shifts from RL's construction to DM's generation, as visualized in Figure 1(a).

This crossing point motivates a principled hybrid integration: leveraging an RL solver to construct a partial solution prefix and subsequently handing over the remaining decisions to a conditional DM for global generation. Three critical questions arise from this hybrid structure: *why* it is beneficial, *when* to trigger the handover, and *how* to implement it on individual trajectories without oracle access.

For the first two, we develop a unified error-scaling framework that aligns the regret dynamics of both paradigms under a common metric. The analysis shows that, under explicit assumptions, a single RL-to-DM handover can achieve strictly lower expected regret than either backbone alone, and that the regret-minimizing trigger is unique, occurring at the marginal-regret balance between the two solvers. For the third, since the optimal trigger is defined at the expected-regret level and cannot be directly evaluated along a single trajectory, we propose HyCO (Figure 1(b)), a plug-and-play hybrid inference algorithm. HyCO combines a single-trigger pipeline with dual-track solution generation and employs a lightweight adaptive trigger based on two complementary proxies: policy entropy, which captures local uncertainty of the RL policy, and RL–DM disagreement, which detects a mismatch between RL's local preference and the DM-induced global prior.

Our contributions are summarized as follows: 1) A unified marginal-regret framework proving that a single RL-to-DM handover achieves strictly lower expected regret than either backbone alone, with a unique regret-minimizing trigger. 2) HyCO, a hybrid inference algorithm with an adaptive dual-signal trigger for detecting the regime shift at the trajectory level. 3) Experiments on TSP, Maximum Independent Set (MIS), and Orienteering Problem (OP) show that HyCO consistently improves over both backbones, while oracle validation and ablations support the effectiveness of the adaptive trigger.

2 Related Work

RL Solvers. RL solvers frame CO problems as sequential decision-making. Following the seminal Pointer Network [15, 4], [5] established the Transformer-based blueprint, later enhanced by techniques for multiple optima [16], solution symmetries [17]. NAR4TSP [18] proposed a non-autoregressive RL algorithm, but with lower solution quality than the autoregressive versions. Subsequent research expanded RL to graph-based generalization [19, 20] and advanced learning paradigms including latent space search [21], hierarchical RL [22], preference optimization [23], and game-theoretic modeling [24]. To enhance performance, recent works integrate RL with heavy decoders [25] or meta-heuristics like genetic algorithms [26] and ant colony sampling [27]. For large-scale COPs, specialized architectures such as Invit [28], UDC [29], and BQ-NCO [30] have been proposed. Despite their efficiency, these solvers inherently suffer from *horizon-wise compounding errors* due to their myopic construction, leading to super-linear regret growth as the problem scale increases.

Heatmap and Diffusion Models for CO. To overcome the sequential limitations of RL solvers, another line of research has focused on non-autoregressive, global DM methods that generate a solution in a single shot, often by producing a "heatmap" of edge or node probabilities [31, 32]. [33] used Graph Convolutional Networks (GCNs) to predict edge inclusion probabilities for TSP. More recently, diffusion models have been adapted for CO. [7] mapped TSP instances to images to be solved by a standard image diffusion model. A more direct approach, DIFUSCO [8], introduced a graph-based diffusion framework that operates on the problem's native graph structure, casting it as a discrete vector generation task. This paradigm has been explored for TSP and extended to other routing problems like VRP [34], while related score-based generative models have been proposed for a broader class of CO problems. While global solvers avoid horizon compounding by capturing global dependencies, they incur a subspace-governed regret that scales with the full solution space dimensionality [14]. Consequently, generating solutions from scratch in high-dimensional settings leads to significant marginal regret, limiting precision unless the effective dimensionality is reduced.

3 Preliminaries

3.1 Sequential Inference with RL Solvers

RL-based solvers formulate CO problems as sequential decision-making processes, where a solution is constructed step by step according to a learned policy. Let $s_k \in \mathcal{S}$ denote the state at step k, and let $\pi(\cdot|s_k)$ be an RL policy that outputs a distribution over feasible actions $\mathcal{A}_k$. At each step, an action is selected according to the policy, for example via greedy decoding:

$$a_k = \arg\max_{a \in \mathcal{A}_k} \pi(a \mid s_k). \tag{1}$$

This sequential construction enables highly efficient inference, requiring only a linear number of forward passes with respect to the solution length. However, decisions made at early steps constrain all subsequent choices. As a result, local errors introduced during the construction process propagate through the remaining horizon, leading to compounding errors and potentially suboptimal solutions.

3.2 Global Inference with Diffusion Models

A DM is a non-autoregressive, generative model that learns to produce a complete solution holistically. For CO problems like TSP, where a solution can be represented by a binary adjacency matrix $A \in \{0, 1\}^{N \times N}$, the model uses a discrete diffusion process. It consists of a denoising network, D_ϕ, trained to reverse a noising process that gradually corrupts the solution by flipping its binary entries. Notably, this process can be conditioned on prior information, such as a given prefix of a tour in a TSP, to guide the generation.

Inference (sampling) starts with a matrix of pure random noise, A_T, and iteratively refines it over T total steps to produce a clean solution, x_0. At each denoising step t (different from the step k in the episode for RL), the denoising network D_ϕ predicts the original clean solution $\hat{x}_0$ based on the current noisy matrix A_t and a conditioning prefix c. This prediction is then used to sample a slightly less noisy matrix A_{t-1}. The reverse step is generally formulated as:

$$A_{t-1} \sim p_\phi(A_{t-1}|A_t, c), t \in \{1, ..., T\}, A_t \in [0, 1]^{N \times N}, \tag{2}$$

where p_ϕ denotes the learned reverse transition distribution. This iterative process allows the model to capture complex global dependencies, leading to high-quality results.

4 Unified Regret Framework and Optimal Trigger Analysis

This section develops a unified theoretical framework for analyzing our RL-to-DM hybrid structure. We begin by aligning RL and DM under a unified regret metric (Definition 4.1), then characterize their contrasting error-scaling behaviors: the DM's regret scales sublinearly with the unsolved dimension (Lemma 4.4), while the RL's regret grows super-linearly with the decision horizon (Lemma 4.5). Building on these lemmas, we prove that under the boundary conditions, there exists a unique optimal trigger step τ^* at which the handover from RL to DM minimizes the total hybrid regret (Theorem 4.6). This optimal step satisfies the marginal-regret balance $r_{\mathrm{RL}}(\tau^*) = r_{\mathrm{DM}}(\tau^*)$, and the resulting hybrid strategy is provably superior to either backbone alone.

Definition 4.1 (Unified Regret Metric)**.** Let $f : \mathcal{X} \to \mathbb{R}$ denote the objective function to be minimized, where $\mathcal{X} \subseteq \{0,1\}^d$ is the d-dimensional discrete decision space. We define the regret $\mathcal{R}(p,q)$ of a solver's output distribution p relative to an optimal target distribution q as the difference in their expected objectives:

$$\mathcal{R}(p,q) = \mathbb{E}_{x\sim p}[f(x)] - \mathbb{E}_{x\sim q}[f(x)], x \in \mathcal{X}. \tag{3}$$

For the RL solver, $q = \delta_{x^*}$ is the Dirac measure at the global optimum x^*. The regret simplifies to the absolute optimality gap: $\mathcal{R}_{RL} = \mathbb{E}_{x\sim\pi}[f(x)] - f(x^*)$. For the conditional DM, given a prefix c, $q(\cdot \mid c)$ denotes the conditional distribution of the optimal completion. The regret instantiates as $\mathcal{R}_{DM}(p, q \mid c) = \mathbb{E}_{x\sim p(\cdot|c)}[f(x)] - \mathbb{E}_{x\sim q(\cdot|c)}[f(x)]$, quantifying the expected suboptimality of the model's output distribution $p(\cdot \mid c)$ relative to $q(\cdot \mid c)$.

To analyze $\mathcal{R}_{DM}$, we require a mild regularity condition on the relaxed objective, i.e. Lipschitz continuity, a condition that is essentially automatic for standard CO problems like TSP and MIS.

Assumption 4.2 (Regularity of Objective Landscape)**.** The continuous relaxation $\tilde{f} : [0,1]^d \to \mathbb{R}$ of the objective function f is *ℓ-Lipschitz continuous*.

Remark 1. *Any C^1 function on the compact domain $[0,1]^d$ is necessarily Lipschitz by the Extreme Value Theorem. In Appendix A.1, we verify that standard CO objectives such as TSP and MIS satisfy this condition under their continuous relaxations.*

With this regularity in hand, we can connect the Kullback-Leibler (KL) divergence minimized during DM training to the optimization regret.

Lemma 4.3 (The Geometry-Statistics Bridge)**.** *Under Assumption 4.2, the regret $\mathcal{R}_{DM}(p,q)$ is upper-bounded by the KL divergence:*

$$\mathcal{R}_{DM}(p,q) \le \ell \cdot \mathcal{C}\sqrt{KL(q\|p)}, \tag{4}$$

where $\mathcal{C}$ is a constant determined by the transport properties of the target distribution q (e.g., its concentration or the domain's diameter).

The proof is in Appendix A.2. Now turn to the regret scaling of the conditional DM. Since the prefix c fixes a subset of decision variables, the model only generates over the remaining unsolved subspace. We show that the regret scales with the square root of the dimension of the unsolved subspace.

Lemma 4.4 (Dimension-Dependent Regret of Conditional DM)**.** *Given a prefix c, let p and q be the generated distribution of a conditional DM and the ground-truth optimal conditional distribution, respectively. Let d_{eff} denote the dimension of the **unsolved subspace**, i.e., the number of decision variables not fixed by the prefix c. Suppose the DM achieves a per-dimension L^2 score-matching error ϵ_{score} on the unsolved subspace, then the regret $\mathcal{R}_{DM}(p, q \mid c)$ under Assumption 4.2 satisfies:*

$$\mathcal{R}_{DM}(p, q \mid c) \le \mathcal{O}\left(\ell \cdot \sqrt{d_{eff}} \cdot \epsilon_{score}\right), \tag{5}$$

where ℓ is the Lipschitz constant of the relaxed objective f.

The proof is in Appendix A.3. We next analyze the regret behavior of the RL solver.

Lemma 4.5 (Super-linear horizon scaling of RL Regret)**.** *For an RL policy π, the error accumulation relative to the **decision horizon** h within a single episode typically exhibits super-linear scaling:*

$$\mathcal{R}_{RL}(h) \sim \mathcal{O}(h^\alpha), \quad \textit{where } \alpha > 1. \tag{6}$$

This scaling captures horizon-wise error accumulation in sequential construction and is motivated by standard finite-horizon RL regret bounds; see Appendix A.4 for details. Having established that the DM's regret scales as $\mathcal{O}(\sqrt{d_{eff}})$ and the RL's regret as $\mathcal{O}(h^{\alpha})$, we observe that under the hybrid inference framework, the two solvers partition the same problem: the RL covers h steps, leaving $d_{eff} = d - h$ variables for the conditional DM. This trade-off naturally gives rise to an optimal handover problem—determining the critical step $\tau^* \in [0, d]$ at which to trigger the conditional DM to minimize the total hybrid regret of the overall CO problem.

Theorem 4.6 (Existence and Uniqueness of the Optimal Trigger Step τ^*)**.** *Given a RL solver and a conditional DM with fixed p and q, let $r_{RL}(h)$ and $r_{DM}(h)$ denote the marginal regret of the RL solver and the DM at decision step h in the overall problem construction, respectively. Define the hybrid regret $\mathcal{L}(\tau) = \int_0^{\tau} r_{RL}(h)\, dh + \int_{\tau}^{d} r_{DM}(h)\, dh$. Under Lemma 4.4 and Lemma 4.5, there exists a **unique** optimal trigger step $\tau^* \in (0, d)$ that minimizes $\mathcal{L}(\tau)$:*

$$r_{RL}(\tau^*) = r_{DM}(\tau^*), \tag{7}$$

if the boundary conditions $r_{RL}(0) < r_{DM}(0)$ and $r_{RL}(d) > r_{DM}(d)$ are satisfied.

Remark 2. *The condition $r_{RL}(0) < r_{DM}(0)$ ensures that RL is initially more efficient than invoking a high-dimensional DM, while $r_{RL}(d) > r_{DM}(d)$ implies that the compounding error of RL eventually exceeds the marginal precision gain of the DM as the unsolved dimensionality d_{eff} vanishes.*

Remark 3 (A General Sufficient Condition of Theorem 4.6)**.** *The specific error-scaling assumptions considered in Lemmas 4.4 and 4.5 provide one sufficient condition for Theorem 4.6, but not necessary conditions. More generally, the conclusion holds whenever the marginal-regret difference $\mathcal{L}'(\tau) := r_{\mathrm{RL}}(\tau) - r_{\mathrm{DM}}(\tau)$ is continuous and strictly increasing on $[0, d]$, with $\mathcal{L}'(0) < 0 < \mathcal{L}'(d)$. The proof of Theorem 4.6 and empirical analysis of regret curves are provided in Appendix A.5.*

A direct consequence of Theorem 4.6 is that **the hybrid framework achieves strictly lower expected regret than either pure RL or pure DM alone**. However, τ^* is defined at the expected-regret level and is not directly observable on a single trajectory, which motivates designing a practical lightweight trigger to estimate the regime transition in Section 5.

5 HyCO: Single-Trigger Hybrid Inference Algorithm

The theoretical analysis in Section 4 yields two insights for algorithm design: (i) a single handover from RL to DM is sufficient to achieve strictly lower regret than either backbone alone, and (ii) the optimal trigger step occurs where the marginal regret of RL equals that of DM. HyCO operationalizes these insights through two components: a single-trigger inference pipeline that executes exactly one handover from RL to DM (Section 5.1), and a lightweight adaptive trigger that detects trajectory-level symptoms of the regime transition as a practical proxy for the theoretical optimum (Section 5.2).

5.1 Single-Trigger inference pipeline

Given a CO instance, HyCO starts with a fast sequential RL solver that constructs a partial solution (prefix) autoregressively. At step k, the RL policy $\pi_\theta(\cdot|s_k)$ outputs a distribution over feasible actions $\mathcal{A}_k$, and a greedy action is selected to extend the prefix. HyCO continuously rolls out the RL solver until a trigger decision is made by a lightweight trigger (Section 5.2). Once the trigger fires at some step k^*, HyCO invokes the conditional diffusion solver once and starts a dual-track generation.

Dual-track solution generation and final selection. At the triggering step k^*, HyCO adopts a dual-track generation strategy to improve candidate diversity under a fixed compute budget: (i) hybrid path correction and (ii) full diffusion proposal generation.

For the hybrid path correction track, the diffusion model is used to *correct the immediate next action* for the hybrid path l_H. Instead of the RL model's greedy choice a_{k^*}, we select a corrected action $a_{k^*}^{DM}$ at step k^* from the best solution in the DM candidate pool $\mathcal{P}_{DM}$. This corrected action is appended to the tour, and the fast RL policy then autoregressively completes the remaining steps.

In parallel, HyCO uses the conditional DM to generate a set of complete, high-quality candidate solutions $\mathcal{P}_{DM}$. Specifically, we take high-probability candidate actions from the RL policy at step k^*, form prefixes with each candidate, and run conditional diffusion completion to obtain full tours.

Finally, HyCO returns the best solution among the hybrid output and the diffusion proposals:

$$l_{\text{out}} = \arg\min_{l \in \{l_H\} \cup \mathcal{P}_{DM}} \text{Cost}(l). \tag{8}$$

The motivation for dual-track generation is practical robustness: Even if the trigger is slightly early or late, the hybrid track provides a low-cost corrected continuation, while the proposal pool increases the probability of recovering a high-quality completion. This parallel RL-based track also serves as a reliability anchor: if the DM fails to produce a feasible or high-quality completion, the hybrid path ensures that the system still returns a valid solution constructed by RL.

5.2 Lightweight trigger as a trigger proxy

HyCO uses a lightweight trigger to decide *when* to invoke conditional global completion. The principle follows from Theorem 4.6: the regret-minimizing handover occurs when the marginal regret of extending the solution prefix with the RL solver matches the marginal regret of invoking the conditional DM for the remaining subproblem; beyond this point, further RL construction is expected to introduce more regret than DM completion. Since this comparison is defined at the expected-regret level and cannot be evaluated directly along a single trajectory, HyCO uses two observable trajectory-level proxies: policy entropy and RL-DM disagreement. Policy entropy serves as an internal reliability signal for RL: high entropy indicates that no feasible action is clearly dominant, making RL's decision less trustworthy. Model disagreement serves as an external alignment signal: large disagreement indicates that the RL policy assigns high probability to candidate next actions that the DM probe considers less consistent with globally coherent solution structure. When either signal becomes large, further RL construction is likely to be less favorable, and HyCO therefore triggers conditional DM completion.

Signal 1: RL policy uncertainty (internal). The first signal measures the RL policy's internal uncertainty via entropy:

$$H(\pi_\theta(\cdot \mid s_k)) = -\sum\nolimits_{a \in \mathcal{A}_k} \pi_\theta(a \mid s_k) \log \pi_\theta(a \mid s_k). \tag{9}$$

High entropy indicates that the policy distribution is flat and no single action is clearly dominant, suggesting a critical decision stage where greedy continuation is more likely to be unstable.

Signal 2: Model disagreement (external). The second signal measures disagreement between the RL policy and a DM-induced global prior. To obtain the diffusion prior efficiently (without running a full sampling trajectory at every step), we probe the conditional diffusion model on a fixed *Top-M* candidate set proposed by RL. Let $\{a_k^{(i)}\}_{i=1}^M$ be the Top-M actions under $\pi_\theta(\cdot|s_k)$, and let $c_k^{(i)} = c_k \oplus a_k^{(i)}$ be its one-step-extended prefixes. We define an energy score $E_\phi(c_k^{(i)})$ from a single-step denoising probe that measures diffusion consistency with the proposed prefix structure:

$$E_\phi(c_k^{(i)}) = -\frac{1}{|\mathcal{E}(c_k^{(i)})|} \sum_{(u,v) \in \mathcal{E}(c_k^{(i)})} \log\left(\sigma\Big(D_\phi(X_{t_{\text{probe}}}, c_k^{(i)})_{uv}\Big)\right), \tag{10}$$

where $\mathcal{E}(c_k^{(i)})$ is the set of edges in the partial tour $c_k^{(i)}$, $\sigma(\cdot)$ is the sigmoid function, and $D_\phi(\cdot)_{uv}$ is the model's predicted logit for edge (u, v). Crucially, this summation acts as a mask: it specifically measures the diffusion model's confidence that the *proposed structural connections* are valid parts of a global solution, ignoring non-connected pairs. We then convert energy values into a diffusion prior via a Boltzmann distribution:

$$q_{DM}(a_k^{(i)} \mid s_k) = \frac{\exp\big(-E_\phi(c_k^{(i)})/\tau_{\text{temp}}\big)}{\sum_{j=1}^M \exp\big(-E_\phi(c_k^{(j)})/\tau_{\text{temp}}\big)}. \tag{11}$$

We then compute the KL divergence between the RL distribution restricted to the same Top-M support and the diffusion prior:

$$D_{KL}(\pi_\theta \,\|\, q_{DM}) = \sum_{i=1}^{M} \pi_\theta(a_k^{(i)} \mid s_k) \, \log \frac{\pi_\theta(a_k^{(i)} \mid s_k)}{q_{DM}(a_k^{(i)} \mid s_k)}. \tag{12}$$

A large divergence suggests that the RL policy's locally preferred action conflicts with the DM's global preference—a regime where invoking conditional global completion often yields gains.

Algorithm 1 HyCO: Hybrid Inference for CO

Require: RL policy π_θ, conditional DM denoiser D_ϕ, thresholds H_{th}, D_{th}, Top-P threshold P_{th}

Initialize hybrid path $l_H \leftarrow ()$, DM proposals $\mathcal{P}_{DM} \leftarrow \emptyset$, state s_0, `triggered` $\leftarrow$ `false`
for $k = 0, \ldots, N-1$ **do**
  if not `triggered` **then**
    $H_k \leftarrow H(\pi_\theta(\cdot \mid s_k))$ *via Eq. (9)*
    Select Top-M actions $\{a^{(i)}\}_{i=1}^{M}$ from $\pi_\theta(\cdot|s_k)$
    Derive diffusion prior $q_{DM}(\cdot|s_k)$ *via Eq. (10, 11)*
    $D_{KL,k} \leftarrow D_{KL}(\pi_\theta \,\|\, q_{DM})$ *via Eq. (12)*
    if $H_k > H_{\text{th}}$ **or** $D_{KL,k} > D_{\text{th}}$ **then**
      Select action set $\{a^*\}$ by cumulative probability threshold P_{th}
      for all selected actions a^* **do**
        Form candidate prefix $c^* \leftarrow s_k \oplus a^*$
        Get full tour $l_{DM} \sim$ DM_Sampler(D_ϕ, c^*)
        $\mathcal{P}_{DM} \leftarrow \mathcal{P}_{DM} \cup \{l_{DM}\}$
      end for
      `triggered` $\leftarrow$ `true`
      Select a_k^{DM} from the best $l \in \mathcal{P}_{DM}$ as the RL action: $a_k \leftarrow a_k^{DM}$
    else
      $a_k \leftarrow \arg\max_a \pi_\theta(a|s_k)$
    end if
  else
    $a_k \leftarrow \arg\max_a \pi_\theta(a|s_k)$
  end if
  Append a_k to hybrid path l_H and update to s_{k+1}
end for
return l_{out}

Triggering rule. In practice, HyCO adopts a simple and robust *or* rule of triggering conditional DM once *either* the RL policy becomes uncertain or the RL–DM disagreement becomes significant:

$$H(\pi_\theta(\cdot \mid s_k)) > H_{\text{th}} \quad \vee \quad D_{KL}(\pi_\theta \| p_\phi) > D_{\text{th}} \tag{13}$$

The *or* rule is easy to tune and empirically stable: high entropy indicates a locally ambiguous decision stage, while a large KL divergence signals a confident but globally inconsistent choice where conditional completion is likely to help.

In all cases, HyCO executes *at most one* trigger: after the conditional diffusion completion is invoked at k^*, no further triggering is allowed, ensuring a predictable compute budget and consistency with the single-trigger structure. Algorithm 1 summarizes the full HyCO inference procedure.

6 Experiments

We evaluate HyCO on standard benchmarks with a primary focus on the 2D Euclidean TSP, MIS problem, and Orienteering Problem (OP) at multiple scales. Our experiments are designed to validate two aspects: (i) Does HyCO consistently improve solution quality over both RL and DM backbones (Section 6.2)? and (ii) Is the trigger step statistically aligned with near-optimal triggering regimes as measured by oracle grid search (Section 6.3)?

6.1 Experimental Settings

Datasets and Baselines. We use standard public datasets for TSP with 50, 100, 500, and 1000 nodes. Problem instances are generated by sampling node coordinates uniformly from the unit square $[0, 1]^2$. We use the Concorde solver [3] to obtain optimal solutions for training our conditional DM and for calculating optimality gaps. We compare our algorithm with a set of methods: exact solver Concorde, heuristic solver LKH-3 [35], and state-of-the-art learning-based approaches, including RL models (AM [5], POMO [16], LEHD [25]), and DM solvers (DIFUSCO [8], T2T [36], CoExpander [34]).

Implementation Details. HyCO uses official pre-trained RL checkpoints and a conditional DM, termed Prefix-Difusco, adapted from DIFUSCO's GNN architecture and trained with masked loss and curriculum learning for prefix-conditioned completion. Full details are provided in Appendix D.

Evaluation Metrics. We report three key metrics: the average tour length (Len), the percentage optimality gap to the exact solver's solutions (Gap), and the average solution time (Time).

6.2 Performance Gains over Backbone Solvers

Small-scale TSP. Table 1 reports the performance on TSP-50 and TSP-100. HyCO achieves near-optimal solutions, consistently outperforming both its RL and conditional DM backbones. Compared to RL solvers, HyCO reduces the optimality gap by over **95%** (from 1.76% / 0.64% to 0.03% on TSP-50). Similarly, HyCO also improves upon the standalone Prefix_Difusco, reducing the gap from 0.27% to 0.03%. Moreover, HyCO outperforms Prefix_Difusco (S), which requires 16 parallel samples, with 4× fewer inference passes—confirming that a well-timed trigger is more effective than brute-force over-sampling. Notably, HyCO surpasses strong full-diffusion baselines like DIFUSCO and remains highly competitive with T2T.

Large-scale TSP. Table 2 evaluates HyCO on TSP-500 and TSP-1000, where long-horizon sequential construction becomes substantially more challenging. With POMO as the RL backbone, HyCO substantially improves over the standalone POMO solver, reducing its optimality gap by 12.06 percentage points on TSP-500 and 10.80 percentage points on TSP-1000, corresponding to relative reductions of 77.2% and 70.2%, respectively. HyCO also reduces the gap of the Prefix_Difusco and Prefix_CoExpander baselines by more than half on both scales, showing that the gain comes from combining an informative RL prefix with conditional diffusion completion rather than from either component alone. The benefit further persists with LEHD, a stronger backbone designed for large-scale neural CO: although the gains are more modest, HyCO still achieves the best overall gaps in the table. These results suggest that HyCO can enhance different RL backbone regimes, including cases where both standalone RL and standalone conditional DM remain far from optimal. Additional 2-opt results in Appendix B.1 show that HyCO maintains its relative advantage after local refinement.

Table 1: Results with Greedy Decoding on TSP-50 and TSP-100. * denotes results that are quoted from previous works. The "Prefix_Difusco (S)" represents the best result obtained by running our Prefix_Difusco model sampling 16 times in parallel.

ALGORITHM	TYPE	TSP-50		TSP-100	
		Len ↓	Gap ↓	Len ↓	Gap ↓
Concorde	Exact	5.69	0.00%	7.76	0.00%
AM*	RL	5.80	1.76%	8.12	4.53%
POMO*	RL	5.73	0.64%	7.84	1.07%
DIFUSCO (T_s=50)	DM	5.71	0.45%	7.85	1.21%
T2T (T_s=50,T_t=15)	DM	5.69	0.07%	7.77	0.20%
Prefix_Difusco	DM	5.70	0.27%	7.81	0.62%
Prefix_Difusco (S)	DM	5.69	0.04%	7.76	0.04%
HyCO (AM)	**Hy**	5.69	**0.03%**	7.76	**0.02%**
HyCO (POMO)	**Hy**	5.69	**0.03%**	7.76	**0.02%**

Beyond TSP, Table 3 shows that HyCO also transfers to MIS. Averaged over RB-LARGE, ER-700-800, and SATLIB, HyCO reduces the optimality gap by 6.12% over the RL baseline LWD. Under the same small candidate budget, it also improves over prefix_Coexpander by an average of 1.75%, and remains better than the heavier prefix_Coexpander (s=64) baseline. A more detailed per-benchmark analysis of MIS is provided in Appendix B.2. OP results in Appendix B.3 show a similar trend on another routing problem: averaged over OP-50/100/200, HyCO reduces the optimality gap by 8.21% over AM and by 3.53% over the DM backbone. Together, these results support that the single-trigger hybrid principle extends beyond TSP to both graph-structured MIS and another routing problem OP.

6.3 Oracle Trigger Validation: Adaptive vs. Instance-wise Best Fixed

To evaluate whether the adaptive trigger can approximate a near-optimal switching time in practice, we compare it with an instance-wise oracle trigger obtained by grid search. We run the same HyCO pipeline over a grid of fixed trigger steps and define the best-performing one as the oracle trigger. Let l_{oracle} and l_{adapt} denote the solutions generated by the oracle and adaptive triggers, respectively. We compare them using the solution gap $\Delta_{\text{cost}} = |\text{Cost}(l_{\text{adapt}}) - \text{Cost}(l_{\text{oracle}})|$ and the trigger-step

Table 2: Results on large-scale TSP problems. RL, SL, G, and S denote Reinforcement Learning, Supervised Learning, Greedy decoding, and Sample decoding, respectively. * indicates the baseline for computing the performance gap. The TIME reports the average inference time per instance.

ALGORITHM	TYPE	TSP-500			TSP-1000		
		Len↓	Gap↓	Time↓	Len↓	Gap↓	Time↓
Concorde	Exact	16.55*	0.00%	37.66m	23.53*	0.00%	6.65h
Gurobi	Exact	16.55	0.00%	45.63h	23.53	0.00%	48h
LKH-3 (default)	Heuristics	16.55	0.00%	46.28m	23.53	0.00%	2.57h
AM	RL+G	20.02	20.99%	0.47s	28.52	21.21%	1.09s
GCN	SL+G	29.72	79.61%	6.67m	43.15	83.38%	28.52m
POMO	RL+G	19.13	15.62%	0.49s	27.15	15.38%	0.94s
DIMES	RL+G	18.93	14.38%	0.97m	27.23	15.73%	2.08m
LEHD	RL/SL+G	16.82	1.64%	1.82s	24.29	3.23%	3.84s
DIFUSCO	SL+G	18.11	9.41%	5.70m	25.68	9.14%	11.5m
T2T	SL+G	17.39	5.09%	4.90m	25.17	8.87%	15.66m
Prefix_Difusco	SL+G	17.92	8.23%	0.14m	25.86	9.91%	0.35m
Prefix_Difusco(S)	SL+S	17.31	4.58%	2.10m	24.96	6.07%	5.94m
Prefix_CoExpander	SL+G	17.61	6.43%	**0.13s**	25.47	8.26%	**0.81s**
HyCO(POMO+Difu)	**RL+SL+G**	17.24	3.56%	1.20m	24.61	4.58%	2.43m
HyCO(LEHD+Difu)	**RL+SL+G**	**16.81**	**1.57%**	0.35m	24.24	3.04%	0.36m
HyCO(LEHD+CoEx)	**RL+SL+G**	**16.81**	**1.57%**	5.32s	**23.79**	**1.09%**	5.89s

Table 3: Performance comparison across RB-LARGE, ER-700-800, and SATLIB benchmarks for the MIS problem. Obj. denotes the average node number of independent sets (higher is better). Gap is the percentage deviation from the optimal solution (lower is better). Time is the average inference time in seconds. S: Sample Decoding. S_{cand} is the candidate number of full DM proposals $\mathcal{P}_{DM}$.

METHOD	RB-LARGE			ER-700-800			SATLIB		
	Obj.↑	Gap↓	Time↓	Obj.↑	Gap↓	Time↓	Obj.↑	Gap↓	Time↓
KaMIS	43.00	0.00%	56.97s	44.97	0.00%	60.75s	425.95	0.00%	24.37s
Gurobi	42.19	1.83%	33.84s	38.78	13.75%	60.49s	425.92	0.01%	13.47s
LWD (RL)	36.67	15.31%	**0.85s**	39.05	13.16%	**0.64s**	421.80	1.04%	0.66s
prefix_Coexpander (Greedy)	40.05	6.84%	1.15s	40.18	10.65%	0.71s	421.84	0.96%	**0.25s**
prefix_Coexpander (s=5)	40.06	6.83%	1.22s	40.85	9.16%	0.96s	424.17	0.42%	0.91s
prefix_Coexpander(s=64)	40.58	5.63%	4.48s	40.91	9.03%	4.03s	424.78	0.27%	4.27s
HyCO (S_{cand}=5)	**40.73**	**5.28%**	2.69s	**42.38**	**5.76%**	4.69s	**425.40**	**0.12%**	6.31s

distance $\Delta_{\text{step}} = |\tau_{\text{adapt}} - \tau_{\text{oracle}}|$. To account for different problem sizes, we also report the normalized distance $\Delta^{\text{r}}_{\text{step}} = \Delta_{\text{step}}/N$, where N is the problem size.

Table 4: Oracle trigger validation on TSP/MIS.

Dataset	Δ_{cost}	Δ_{cost}/f^*	Δ_{step}	$\Delta^{\text{r}}_{\text{step}}$
TSP-100	0.0028	0.036%	1.48	1.48%
TSP-500	0.3424	2.057%	3.23	0.65%
SATLIB	0.3812	0.089%	22.60	1.77%
ER 700–800	1.0460	2.330%	18.62	2.48%

Results Analysis. Table 4 shows that the adaptive trigger closely tracks the instance-wise oracle in trigger location, with normalized step distances below 2.5% across all benchmarks. The resulting solution quality is also close to the oracle: the relative cost difference is only 0.036% on TSP-100, 2.057% on TSP-500, 0.089% on SATLIB, and 2.330% on ER-700–800. The slightly larger deviations on TSP-500 and ER-700–800 suggest that larger or structurally more complex instances can be more sensitive to small trigger-step shifts, even when the normalized trigger distance remains small. These results indicate that the adaptive trigger identifies a near-oracle switching region in practice, supporting entropy and RL–DM disagreement as effective trajectory-level proxies for adaptive triggering.

Further experiments in Appendix B.4 compare adaptive triggering with validation-selected fixed-time schedules, while Appendix B.5 reports trigger statistics across datasets and problem scales. Appendix C presents ablation studies on trigger-threshold calibration, energy-probe and decoding

hyperparameters, the dual-signal trigger, the complementarity of the two output tracks, and the trade-off between solution quality and inference time under single and multiple triggering.

7 Conclusion

We introduced HyCO, a hybrid neural solver that combines sequential RL construction with conditional DM completion for CO. Motivated by the complementary regret behavior of RL and DM backbones, our unified regret analysis shows that a single RL-to-DM handover can achieve strictly lower expected regret than either backbone alone, and that the regret-minimizing handover occurs at the marginal-regret balance between the two solvers. Since this marginal-regret balance is not directly observable on individual trajectories, HyCO uses policy entropy and model disagreement as lightweight trajectory-level proxies for detecting the corresponding regime transition in practice. Experiments on TSP, MIS, and OP demonstrate consistent improvements over both backbones, while oracle validation and ablations support the effectiveness of the adaptive dual-signal trigger. While our theoretical analysis focuses on RL and DM backbones, the core insight—complementary error scaling between sequential and global solvers—may generalize to other solver pairings with analogous properties, which we leave for future work. More broadly, we hope this work opens a new direction for neural combinatorial optimization through adaptive collaboration among solvers with complementary strengths.

Acknowledgements and Disclosures

Funding. The work done by Y. Li and H. Chen was supported by the National Science Foundation under IIS-2348405, the Coastal Virginia Center for Cyber Innovation (COVA CCI), Commonwealth Cyber Initiative (CCI), the William & Mary Faculty Research Grant, and generous gifts from NVIDIA and Amazon. The work done by D. Yang and Y. Xiong was supported by the Coastal Virginia Center for Cyber Innovation (COVA CCI) and the Commonwealth Cyber Initiative (CCI).

Competing interests. The authors declare no financial relationships or competing interests outside the submitted work that could reasonably be perceived as influencing this work.

References

[1] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Mathematical Sciences Series. W. H. Freeman, 1979.

[2] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

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

[4] Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. In *ICLR*, 2017.

[5] Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *ICLR*, 2019.

[6] Nina Mazyavkina, Sergey Sviridov, Sergei Ivanov, and Evgeny Burnaev. Reinforcement learning for combinatorial optimization: A survey. *Computers & Operations Research*, 134:105400, 2021.

[7] Alexandros Graikos, Nikolay Malkin, Nebojsa Jojic, and Dimitris Samaras. Diffusion models as plug-and-play priors. In *NeurIPS*, pages 14715–14728, 2022.

[8] Zhiqing Sun and Yiming Yang. DIFUSCO: Graph-based diffusion solvers for combinatorial optimization. In *NeurIPS*, pages 3706–3731, 2023.

[9] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *AISTATS*, pages 661–668, 2010.

[10] Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, pages 627–635, 2011.

[11] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael Jordan. Is q-learning provably efficient? In *NeurIPS*, volume 31, 2018.

[12] Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *COLT*, pages 2137–2143, 2020.

[13] Grigoris Velegkas, Zhuoran Yang, and Amin Karbasi. Reinforcement learning with logarithmic regret and policy switches. In *NeurIPS*, pages 36040–36053, 2022.

[14] Joe Benton, Valentin De Bortoli, Arnaud Doucet, and George Deligiannidis. Nearly d-linear convergence bounds for diffusion models via stochastic localization. In *ICLR*, 2024.

[15] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *NeurIPS*, 2015.

[16] Yeong-Dae Kwon, Juhan Choo, Bumm Kim, Iljoo Yoon, Young-Joon Gwon, and Seung-won Min. POMO: Policy optimization with multiple optima for reinforcement learning. In *NeurIPS*, pages 21188–21198, 2020.

[17] Minsu Kim, Jinkyoo Park, et al. Sym-NCO: Leveraging symmetricity for neural combinatorial optimization. In *NeurIPS*, pages 1936–1949, 2022.

[18] Yubin Xiao, Di Wang, Boyang Li, Huanhuan Chen, Wei Pang, Xuan Wu, Hao Li, Dong Xu, Yanchun Liang, and You Zhou. Reinforcement learning-based nonautoregressive solver for traveling salesman problems. *IEEE Transactions on Neural Networks and Learning Systems*, 36(7):13402–13416, 2025.

[19] 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.

[20] Haipeng Chen, Wei Qiu, Han-Ching Ou, Bo An, and Milind Tambe. Contingency-aware influence maximization: A reinforcement learning approach. In *UAI*, pages 1535–1545, 2021.

[21] Felix Chalumeau, Shikha Surana, Clément Bonnet, Nathan Grinsztajn, Arnu Pretorius, Alexandre Laterre, and Tom Barrett. Combinatorial optimization with policy adaptation using latent space search. In *NeurIPS*, pages 7947–7959, 2023.

[22] Xinsong Feng, Zihan Yu, Yanhai Xiong, and Haipeng Chen. Sequential stochastic combinatorial optimization using hierarchal reinforcement learning. In *ICLR*, 2025.

[23] Mingjun Pan, Guanquan Lin, You-Wei Luo, Bin Zhu, Zhien Dai, Lijun Sun, and Chun Yuan. Preference optimization for combinatorial optimization problems. In *ICML*, 2025.

[24] Yuheng Li, Panpan Wang, and Haipeng Chen. Can reinforcement learning solve asymmetric combinatorial-continuous zero-sum games? In *ICLR*, 2025.

[25] Fu Luo, Xi Lin, Fei Liu, Qingfu Zhang, and Zhenkun Wang. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. In *NeurIPS*, pages 8845–8864, 2023.

[26] Hyeonah Kim, Sanghyeok Choi, Jiwoo Son, Jinkyoo Park, and Changhyun Kwon. Neural genetic search in discrete spaces. In *ICML*, 2025.

[27] Minsu Kim, Sanghyeok Choi, Hyeonah Kim, Jiwoo Son, Jinkyoo Park, and Yoshua Bengio. Ant colony sampling with gflownets for combinatorial optimization. In *AISTATS*, pages 469–477, 2025.

[28] Han Fang, Zhihao Song, Paul Weng, and Yutong Ban. Invit: A generalizable routing problem solver with invariant nested view transformer. In *ICML*, pages 12973–12992, 2024.

[29] Zhi Zheng, Changliang Zhou, Tong Xialiang, Mingxuan Yuan, and Zhenkun Wang. Udc: A unified neural divide-and-conquer framework for large-scale combinatorial optimization problems. In *NeurIPS*, pages 6081–6125, 2024.

[30] Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. BQ-NCO: Bisimulation quotienting for efficient neural combinatorial optimization. In *NeurIPS*, pages 77416–77429, 2023.

[31] Zhuwen Li, Qifeng Chen, and Vladlen Koltun. Combinatorial optimization with graph convolutional networks and guided tree search. In *NeurIPS*, pages 562–572, 2018.

[32] Zhang-Hua Fu, Kai-Bin Qiu, and Hongyuan Zha. Generalize a small pre-trained model to arbitrarily large tsp instances. In *AAAI*, pages 7474–7482, 2021.

[33] Chaitanya K. Joshi, Thomas Laurent, and Xavier Bresson. An efficient graph convolutional network technique for the travelling salesman problem. In *INFORMS*, 2019.

[34] Jiale Ma, Wenzheng Pan, Yang Li, and Junchi Yan. COExpander: Adaptive solution expansion for combinatorial optimization. In *ICML*, 2025.

[35] Keld Helsgaun. An extension of the lin-kernighan-helsgaun tsp solver for constrained traveling salesman and vehicle routing problems. *Roskilde: Roskilde University*, 12:966–980, 2017.

[36] Yang Li, Jinpei Guo, Runzhong Wang, and Junchi Yan. T2t: From distribution learning in training to gradient search in testing for combinatorial optimization. In *NeurIPS*, pages 50020–50040, 2023.

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

[38] Xavier Bresson and Thomas Laurent. The transformer network for the traveling salesman problem. *arXiv preprint arXiv:2103.03012*, 2021.

[39] Federico Berto, Chuanbo Hua, Junyoung Park, Laurin Luttmann, Yining Ma, Fanchen Bu, Jiarui Wang, Haoran Ye, Minsu Kim, Sanghyeok Choi, et al. Rl4co: an extensive reinforcement learning for combinatorial optimization benchmark. In *KDD*, pages 5278–5289, 2025.

Technical Appendices and Supplementary Material

A Theoretical Proof **14**

A.1 On the Validity and Applicability of Assumption 4.2 14

A.2 Proof of Lemma 4.3 14

A.3 Proof of Lemma 4.4: Dimension-Dependent Regret of Conditional DM 15

A.4 Justification of Lemma 4.5: Super-linear RL Horizon Scaling 16

A.5 Proof and Empirical Regret Analysis of Theorem 4.6: Existence and Uniqueness of the Optimal Trigger Step τ^* 16

B Additional Experimental Results **17**

B.1 TSP Performance Results 17

B.2 MIS Performance Results 19

B.3 OP Performance Results 20

B.4 Adaptive Trigger vs. Fixed-time Triggering 20

B.5 Cross-Dataset Trigger Statistics 21

C Ablation Study of HyCO **22**

C.1 Hyperparameter Sensitivity of Triggering and DM Exploration 22

C.2 Necessity of Dual-Signal Trigger Design 24

C.3 Complementarity of the Two Output Tracks 24

C.4 Effect of Multiple Triggering on Solution Quality and Inference Time 25

D Implementation Details of Prefix_Difusco and RL models **26**

D.1 Model Architecture 26

D.2 Training Procedure 26

D.3 Decoding Algorithms 27

D.4 Hyperparameters 28

E Limitations and Broader Impacts **28**

A Theoretical Proof

A.1 On the Validity and Applicability of Assumption 4.2

To bridge the gap between discrete combinatorial structures and the continuous dynamics of Diffusion Models (DMs), we justify the L-Lipschitz continuity of the objective function f under its continuous relaxation. This regularity is a fundamental requirement for the subsequent regret analysis.

In DMs, the discrete objective function $f : \{0,1\}^d \to \mathbb{R}$ is extended to the continuous unit hypercube $\mathcal{X} = [0,1]^d$. For any combinatorial problem, there exists a canonical **multilinear extension** $\tilde{f} : [0,1]^d \to \mathbb{R}$ defined as:

$$\tilde{f}(x) = \sum_{S \subseteq \{1,\dots,d\}} f(S) \prod_{i \in S} x_i \prod_{j \notin S} (1 - x_j), \quad x \in \mathcal{X}, \tag{14}$$

where S denotes a discrete candidate solution (e.g., a specific subset of edges in TSP), and x_i represents the marginal probability of element i being included in the solution, often represented as a probabilistic "heatmap". Mathematically, $\tilde{f}(x)$ represents the expected objective value $\mathbb{E}_{S \sim x}[f(S)]$ under independent Bernoulli sampling. Since $\tilde{f}$ is a polynomial in x, it is continuously differentiable (C^1) on the compact set $\mathcal{X}$.

By the Extreme Value Theorem, any continuously differentiable function on a compact set has a bounded gradient. Let $L = \sup_{x \in \mathcal{X}} \|\nabla \tilde{f}(x)\|$. Then, for any $x, y \in \mathcal{X}$, the Mean Value Theorem implies:

$$|\tilde{f}(x) - \tilde{f}(y)| \leq L \|x - y\| \tag{15}$$

This confirms that the relaxed objective is L-Lipschitz continuous. We specifically highlight its applicability to the benchmarks considered:

- Linear Objectives (e.g., TSP): The objective is $f(x) = \langle W, x \rangle$. The gradient $\nabla f = W$ is a constant edge-weight matrix, thus $L = \|W\|_F$ is finite and independent of x.
- Quadratic Objectives (e.g., MIS): Formulated as $f(x) = x^T Q x$, the gradient $\nabla f = 2Qx$ is linear in x. On the bounded domain $[0,1]^d$, the spectral properties of the graph matrix Q ensure the gradient remains bounded, satisfying the L-Lipschitz condition.

The L-Lipschitz property is the minimal requirement to satisfy the Kantorovich-Rubinstein duality. It allows us to bound the optimization regret by the Wasserstein-1 distance (W_1), which is subsequently connected to the KL divergence matching the DM training objective via Talagrand's inequality in Lemma 4.3.

A.2 Proof of Lemma 4.3

Proof. We establish this bound by decomposing the optimization regret into a geometric component (landscape regularity) and a statistical component (transportation cost).

The regret $\mathcal{R}(p, q)$ is defined as the difference in expected values of the objective f under distributions p and q. Given that the continuous relaxation $\tilde{f}$ is ℓ-Lipschitz (Assumption 4.2), we invoke the Kantorovich-Rubinstein duality for the W_1 distance:

$$\mathcal{R}(p, q) = |\mathbb{E}_{x \sim p}[\tilde{f}(x)] - \mathbb{E}_{x \sim q}[\tilde{f}(x)]| \leq \sup_{|\phi|_{Lip} \leq \ell} \left| \int \phi dp - \int \phi dq \right| = \ell \cdot W_1(p, q), \tag{16}$$

where $\text{Lip}_\ell(\mathcal{X}) = \{\phi : \mathcal{X} \to \mathbb{R} \mid |\phi|_{Lip} \leq \ell\}$ denotes the set of all functions with Lipschitz constant at most ℓ. By the definition of the W_1 distance, the supremum on the right-hand side is exactly $\ell \cdot W_1(p, q)$. This step translates the optimization gap into a geometric distance between distributions.

Then, to relate the geometric distance W_1 to the information-theoretic divergence $KL(q\|p)$, we consider two distinct regimes to ensure the universality of the lemma:

- **General Case (Pinsker's Inequality):** On a compact solution manifold $\mathcal{X}$ with diameter $D = \sup_{x,y \in \mathcal{X}} \|x - y\|$, the W_1 distance is bounded by the Total Variation (TV) distance:

$W_1(p,q) = W_1(q,p) \leq D \cdot TV(q,p)$. By Pinsker's inequality, $TV(q,p) \leq \sqrt{\frac{1}{2}KL(q||p)}$, yielding:

$$W_1(p,q) \leq \frac{D}{\sqrt{2}}\sqrt{KL(q \parallel p)}. \tag{17}$$

This bound holds for any arbitrary target distribution q on a bounded domain, justifying the lemma's applicability to general NCO tasks beyond diffusion-based priors.

- **Regular Case (Talagrand's T_2 Inequality):** For target distributions q that exhibit stronger concentration (e.g., Gaussian-smoothed Gibbs measures used in DM), we assume q satisfies a Log-Sobolev Inequality with constant $1/\sigma^2$. This implies the Talagrand's T_2 inequality: $W_1(p,q) \leq W_2(p,q) = W_2(q,p) \leq \sqrt{2\sigma^2 KL(q\|p)}$.

By defining $\mathcal{C} = \min\{\frac{D}{\sqrt{2}}, \sqrt{2\sigma^2}\}$ (or more generally, treating $\mathcal{C}$ as a constant capturing the domain diameter or concentration radius), we obtain:

$$\mathcal{R}(p,q) \leq \ell \cdot \mathcal{C}\sqrt{KL(q||p)}. \tag{18}$$

□

A.3 Proof of Lemma 4.4: Dimension-Dependent Regret of Conditional DM

Proof. This proof establishes that the optimization regret of a conditional DM is governed by the intrinsic dimensionality of the unsolved decision space. We provide a rigorous derivation by projecting the dynamics onto the free subspace and utilizing the conditional covariance dissipation identity.

In our task, the conditioning $Mx = c$ restricts the generative process to an affine manifold $\mathcal{V}_c := \{x \in \mathbb{R}^d \mid Mx = c\}$, where c is the prefix (partial solution). Let $\mathbf{P} = \mathbf{I} - \mathbf{M}$ be the orthogonal projection onto the unsolved (free) subspace. We decompose the state as $x = \mathbf{M}x + \mathbf{P}x$. Since $\mathbf{M}x$ is fixed to the constant vector c, the forward SDE (e.g., Variance Preserving or Variance Exploding) admits a degenerate transition kernel $q_t(x|c) = \nu_t(z|c) \otimes \delta_c(u)$, where $z = \mathbf{P}x \in \mathbb{R}^{d_{eff}}$ and $d_{eff} = \text{rank}(\mathbf{P})$.

The conditional score function $\nabla \log q_t(x|c)$ is only well-defined on the subspace $\text{Im}(\mathbf{P})$, and its components in $\text{Im}(\mathbf{M})$ vanish. Consequently, the reverse SDE is strictly confined to $\mathcal{V}_c$:

$$dx = \{f(x,t) - g(t)^2 \nabla \log q_t(x|c)\}dt + g(t)d\bar{w} \tag{19}$$

where the stochastic evolution and score estimation error ϵ_{score} are restricted to d_{eff} degrees of freedom, effectively reducing the ambient complexity from d to d_{eff}.

To bound the KL divergence on the manifold, we extend the nearly d-linear framework [14] to the conditional case. Like Assumption 1 in [14], we assume the model achieves a per-dimension L^2 error ϵ_{score} specifically on the unsolved manifold $\mathcal{V}_c$.

A critical pillar of this proof is the Conditional Covariance Dissipation Identity. Let $\mathbf{\Sigma}_t := \text{Cov}(X_0|X_t, c)$ be the conditional posterior covariance. Following the Tweedie relation, the score function's Hessian is related to $\mathbf{\Sigma}_t$. For a general class of linear SDEs (not limited to OU), the expected covariance evolves as:

$$\frac{\sigma_t^3}{2\dot{\sigma}_t}\frac{d}{dt}\mathbb{E}[\mathbf{\Sigma}_t|c] = -\mathbb{E}[\mathbf{\Sigma}_t^2|c] \tag{20}$$

where σ_t^2 is the noise schedule. While [14] derived this for the unconditional case, the identity holds on the subspace $\text{Im}(\mathbf{P})$ due to the orthogonality of $\mathbf{M}$ and $\mathbf{P}$. Since $\text{rank}(\mathbf{\Sigma}_t) = d_{eff}$, the trace $\text{Tr}(\mathbf{\Sigma}_t)$—which governs the discretization drift—scales linearly with d_{eff}.

To avoid the score singularity at $t = 0$, we consider the divergence between the generated distribution p and the Gaussian-smoothed target q_δ at early-stopping time $\delta > 0$. Integrating over the time horizon T, the KL divergence scales as:

$$D_{KL}(q_\delta \| p \mid c) \leq \mathcal{C}[\epsilon_{score}^2 + \kappa^2 d_{eff} N + \kappa d_{eff} T + d_{eff} e^{-2T}] = \mathcal{O}(d_{eff} \cdot \epsilon_{score}^2) \tag{21}$$

The initialization and discretization error components are fundamentally linear in the number of unsolved variables d_{eff}.

Finally, we invoke Lemma 4.3 (The Geometry-Statistics Bridge). Given the ℓ-Lipschitz continuity of the objective f, the regret is bounded by the W_1 distance, which is in turn controlled by the KL divergence via the transportation-cost inequality. Substituting the d_{eff}-linear KL bound into the bridge inequality yields:

$$\mathcal{R}_{DM}(p, q \mid c) \leq \ell \cdot \mathcal{C}\sqrt{D_{KL}(q_\delta \| p \mid c)} \leq \mathcal{O}(\ell \cdot \sqrt{d_{eff}} \cdot \epsilon_{score}) \tag{22}$$

where we treat the discrepancy between q and q_δ as a negligible additive bias for $\delta \ll 1$. □

A.4 Justification of Lemma 4.5: Super-linear RL Horizon Scaling

Lemma 4.5 models the accumulated regret of an autoregressive RL construction policy as super-linear in the decision horizon. This scaling is motivated by standard regret bounds in finite-horizon RL. In tabular episodic MDPs, Q-learning with UCB exploration achieves a regret bound of the form

$$\widetilde{O}\Big(\sqrt{H^3SAT}\Big),$$

where H is the episode horizon, S and A are the numbers of states and actions, and T is the total number of interaction steps [11]. For RL with linear function approximation, related analyses obtain regret bounds of the form

$$\widetilde{O}\Big(\sqrt{d^3H^3T}\Big),$$

where d denotes the feature dimension [12, 13]. In both cases, the dependence on the horizon contains an $H^{3/2}$ factor, which is super-linear in H. More broadly, super-linear horizon dependence is a common feature of finite-horizon RL regret analyses. These bounds provide theoretical support for the horizon-scaling model in Lemma 4.5, although they do not directly characterize neural CO solvers.

Autoregressive neural CO solvers such as AM and POMO instantiate sequential construction policies: a solution is produced through a sequence of decisions whose horizon grows with the problem size. Existing finite-horizon RL bounds do not directly characterize these neural policy-gradient solvers, but they provide a principled motivation for modeling accumulated RL regret as super-linear in the construction horizon. This modeling choice is also consistent with the empirical behavior of autoregressive RL solvers on large-scale CO instances, where solution quality often degrades as the number of construction steps increases.

A.5 Proof and Empirical Regret Analysis of Theorem 4.6: Existence and Uniqueness of the Optimal Trigger Step τ^*

Proof. This proof analyzes the hybrid regret $\mathcal{L}(\tau)$ to establish the uniqueness and global optimality of the trigger step τ^*.

Based on the definition of $\mathcal{L}(\tau) = \int_0^\tau r_{\text{RL}}(h)\,dh + \int_\tau^d r_{\text{DM}}(h)\,dh$, we can get that the derivative of the hybrid regret with respect to the trigger step is:

$$\mathcal{L}'(\tau) = r_{\text{RL}}(\tau) - r_{\text{DM}}(\tau).$$

According to Lemma 4.5, $r_{RL}(h) = \frac{d}{dh}\mathcal{R}_{\text{RL}}(h) \sim \mathcal{O}(h^{\alpha-1})$ with $\alpha - 1 > 0$, which is strictly increasing in h, i.e. $r_{\text{RL}}(h) \geq 0$.

For the regret of DM, it satisfies $R_{\text{DM}}(d) \leq \mathcal{O}(\sqrt{d})$ by Lemma 4.4, where $R_{\text{DM}}(d)$ means the regret of DM (fixed distribution p and q so we ignore its signal in the regret) with d-dimensional decision variables. Since $\sqrt{d}$ is concave in d, $R_{\text{DM}}(d)$ exhibits diminishing marginal returns as the problem size d increases. In the hybrid framework, the regret of conditional DM facing a problem of size $d - \tau$ is calculated by $\int_\tau^d r_{\text{DM}}(h)\,dh = \int_0^d r_{\text{DM}}(h)\,dh - \int_0^\tau r_{\text{DM}}(h)\,dh = R_{\text{DM}}(d) - R_{\text{DM}}(\tau)$. Due to the decreasing marginal regret of R_{DM}, we can get $r_{\text{DM}}(\tau)$ is non-increasing.

Since $r_{\text{RL}}(\tau)$ is strictly increasing and $r_{\text{DM}}(\tau)$ is non-increasing, their difference $\mathcal{L}'(\tau) = r_{\text{RL}}(\tau) - r_{\text{DM}}(\tau)$ is strictly increasing. The boundary conditions $r_{\text{RL}}(0) < r_{\text{DM}}(0)$ and $r_{\text{RL}}(d) > r_{\text{DM}}(d)$ imply $\mathcal{L}'(0) < 0$ and $\mathcal{L}'(d) > 0$. Since $\mathcal{L}'(\tau)$ is continuous and strictly increasing, the Intermediate Value Theorem guarantees exactly one $\tau^* \in (0, d)$ such that $\mathcal{L}'(\tau^*) = 0$, i.e., $r_{\text{RL}}(\tau^*) = r_{\text{DM}}(\tau^*)$. Since $\mathcal{L}'(\tau) < 0$ for $\tau < \tau^*$ and $\mathcal{L}'(\tau) > 0$ for $\tau > \tau^*$, this τ^* is the unique global minimum of $\mathcal{L}(\tau)$.

Specially, since $\mathcal{L}'(\tau)$ is strictly increasing with $\mathcal{L}'(0) < 0$ and $\mathcal{L}'(d) > 0$, $\mathcal{L}$ is strictly decreasing on $[0, \tau^*)$ and strictly increasing on $(\tau^*, d]$. This implies $\mathcal{L}(\tau^*) < \mathcal{L}(0)$ and $\mathcal{L}(\tau^*) < \mathcal{L}(d)$, i.e., the hybrid framework achieves strictly lower expected regret than either pure RL or pure DM alone. □

Generality of the sufficient conditions. The proof above establishes the conclusion through the continuity, strict monotonicity, and boundary sign change of the marginal-regret difference $\mathcal{L}'(\tau)$. The particular scaling forms considered in Lemmas 4.4 and 4.5 are not necessary for the conclusion. In particular, $\mathcal{L}'(\tau)$ remains strictly increasing when $r_{\mathrm{RL}}(\tau)$ is non-decreasing and $r_{\mathrm{DM}}(\tau)$ is strictly decreasing. This permits regret profiles beyond the specific scaling forms considered above, including linear RL regret accumulation, provided that the stated boundary conditions also hold. The applicability of the result is conditional on these structural properties of the regret profiles.

Empirical examination of the regret curves. We examine the regret profiles of the trained solvers on 128 TSP-500 instances. For the RL POMO solver, we greedily construct prefixes of different lengths h and solve each remaining prefix-constrained subproblem to optimality. We estimate the accumulated regret as the difference between the optimal objective under the constructed prefix and the unconstrained optimal objective, and fit a power-law model $R_{\mathrm{RL}}(h) \approx ah^{\alpha}$ to the measurements for each instance. Fitting $R_{\mathrm{RL}}(h) \propto h^{\alpha}$ yields per-instance exponents ranging from 0.93 to 1.16, with a median of $\hat{\alpha} = 1.02$. These results indicate approximately linear regret accumulation over the measured range, with variation across individual instances.

For the DM solver, we condition the trained model on progressively longer prefixes taken from an optimal tour. This controls prefix quality and isolates the effect of the remaining free dimension d_{eff}. We measure the completion regret relative to the optimal tour and fit an exponential model to its dependence on d_{eff}. Over the measured range, the completion regret increases approximately exponentially with the remaining dimension, $R_{\mathrm{DM}}(d_{\mathrm{eff}}) \propto \exp(\lambda d_{\mathrm{eff}}), \lambda \approx 0.019$. Equivalently, the fitted completion regret decreases approximately exponentially as the prefix grows and the remaining subspace shrinks. This finite-range empirical fit supports the monotonic dimension dependence predicted by Lemma 4.4, but it is not intended as a tight characterization of the theoretical upper bound. Importantly, these empirical forms still preserve the relative monotonicity required by Theorem 4.6: $\mathcal{L}'(\tau)$ is strictly increasing.

Overall, Theorem 4.6 characterizes the unique-handover result under explicit conditions. Lemmas 4.4 and 4.5 provide one sufficient route to its conclusion through specific RL and DM scaling forms, whereas the proof itself relies on the more general relative one-crossing condition. The empirical curves exhibit an alternative instantiation of the required structure: an approximately linear RL regret curve with a non-decreasing marginal trend, together with an exponentially decreasing conditional-DM marginal regret. For the fitted aggregate trends, this makes $\mathcal{L}'(\tau)$ increasing, which is consistent with the one-crossing condition and supports the same unique-handover mechanism under the stated boundary conditions.

B Additional Experimental Results

For the main performance comparisons and the following result tables in Appendix B, we evaluate on 1,280 test instances each for TSP-50 and TSP-100, and 128 instances each for TSP-500 and TSP-1000. For MIS, the RB-LARGE, ER-700–800, and SATLIB test sets contain 500, 128, and 500 instances, respectively. For OP, we evaluate on 500 test instances at each problem size: OP-50, OP-100, and OP-200. These counts refer to distinct problem instances and are separate from the number of candidate solutions generated per instance.

B.1 TSP Performance Results

For TSP-50/100, we omit wall-clock time because several baselines are quoted from prior work under heterogeneous hardware settings, while greedy inference for small instances is already in the millisecond-scale regime and can be dominated by hardware, batching, and implementation overhead. We therefore focus time reporting on large-scale TSP, where computational differences are more meaningful.

Impact of Local Search (2-opt). To investigate whether the improvements of HyCO stem merely from local refinements or from a fundamentally better global solution structure, we evaluated the

Table 5: Results with **Greedy Decoding** on TSP-50 and TSP-100. RL: Reinforcement Learning, SL: Supervised Learning, G: Greedy Decoding, S: Sample Decoding. * denotes results that are quoted from previous works. The "Prefix_Difusco (S)" represents the best result obtained by running our Prefix_Difusco model sampling 16 times in parallel.

ALGORITHM	TYPE	TSP-50		TSP-100	
		Len ↓	Gap ↓	Len ↓	Gap ↓
Concorde [3]	Exact	5.69	0.00%	7.76	0.00%
2Opt [37]	Heuristics	5.86	2.95%	8.03	3.54%
AM* [5]	RL+G	5.80	1.76%	8.12	4.53%
GCN* [33]	SL+G	5.87	3.10%	8.41	8.38%
Transformer* [38]	RL+G	5.71	0.31%	7.88	1.42%
POMO* [16]	RL+G	5.73	0.64%	7.84	1.07%
Sym-NCO* [17]	RL+G	5.73	0.64%	7.84	0.94%
Image Diffusion* [7]	SL+G	5.76	1.23%	7.92	2.11%
DIFUSCO (T_s=50) [8]	SL+G	5.71	0.45%	7.85	1.21%
T2T (T_s=50,T_t=15) [36]	SL+G	5.69	0.07%	7.77	0.20%
Prefix_Difusco	SL+G	5.70	0.27%	7.81	0.62%
Prefix_Difusco (S)	SL+S	5.69	0.04%	7.76	0.04%
HyCO (AM)	**RL+SL+G**	5.69	**0.03%**	7.76	**0.02%**
HyCO (POMO)	**RL+SL+G**	5.69	**0.03%**	7.76	**0.02%**
AM	RL+G+2OPT	5.77	1.41%	8.02	3.32%
GCN	SL+G+2OPT	5.77	1.40%	8.01	3.21%
Transformer	RL+G+2OPT	5.70	1.06%	7.96	1.89%
POMO	SL+G+2OPT	5.73	0.63%	7.91	1.62%
Sym-NCO	SL+G+2OPT	5.73	0.64%	7.90	0.76%
DIFUSCO	SL+G+2OPT	5.69	0.09%	7.78	0.22%
T2T	SL+G+2OPT	5.69	**0.02%**	7.76	0.06%
HyCO (POMO)	**RL+SL+G+2OPT**	5.69	0.03%	7.76	**0.01%**

performance when a 2-opt local search post-processing step [37] is applied. As indicated in our experimental logs, while 2-opt consistently improves the objective values for all constructive methods, the relative advantage of HyCO persists. On large-scale instances (TSP-1000), even after exhaustive 2-opt refinement, the standard RL baselines (e.g., POMO+2opt) typically fail to reach the solution quality of HyCO. This observation implies that the baseline solvers often get trapped in poor local basins due to early-stage compounding errors, which local search cannot rectify. In contrast, HyCO utilizes the diffusion model to bridge the critical regime shift, producing a globally superior solution skeleton that is more amenable to local refinement. This confirms that HyCO contributes to *global consistency* rather than just local smoothness.

Sample Efficiency and Inference Budget. A critical comparison in our results lies between HyCO and the brute-force conditional baseline, Prefix_Difusco (S). The latter represents a standard sampling approach where the model generates 16 independent completions for a fixed prefix and selects the best one. Remarkably, HyCO achieves solution quality that is comparable to or better than this baseline despite using a single guided completion ($S = 4$). For instance, on TSP-100, HyCO (0.02% gap) outperforms the 16-sample baseline (0.04% gap). This result reinforces the core premise of our theory: the *triggering step* is more consequential than the *quantity* of samples. By adaptively identifying the optimal triggering time window, HyCO eliminates the need for wasteful over-sampling, achieving superior efficiency with **4× fewer inference passes**. This structural advantage suggests that HyCO effectively locates the optimal low-regret subspace, whereas blind over-sampling struggles to compensate for a suboptimal triggering time.

Plug-and-Play Versatility. To validate the universality of our framework, we extended our evaluation by pairing HyCO with three distinct sequential architectures: the standard autoregressive AM, the symmetry-enhanced POMO, and the divide-and-conquer LEHD. HyCO yields consistent performance gains across all backbones, regardless of their intrinsic strength. For weaker solvers like AM, HyCO acts as a revitalizing mechanism, reducing the optimality gap on TSP-1000 from over 20% to about

5% by correcting severe horizon-wise error accumulation. Conversely, for state-of-the-art constructive solvers like LEHD, HyCO provides further refinement, pushing the limits of solution quality. This consistent improvement confirms that HyCO serves as a general-purpose, test-time inference wrapper that addresses the fundamental uncertainty accumulation inherent to sequential construction, rather than being tailored to the idiosyncrasies of a specific policy.

Table 6: Results on large-scale TSP problems. RL, SL, AS, G, and S denote Reinforcement Learning, Supervised Learning, Active Search, Greedy decoding, and Sample decoding, respectively. Len means the average tour length. * indicates the baseline for computing the performance gap. The Time column reports the average inference time per instance.

ALGORITHM	**TYPE**	**TSP-500**			**TSP-1000**		
		Len↓	Gap↓	Time↓	Len↓	Gap↓	Time↓
Concorde	Exact	16.55*	0.00%	37.66m	23.53*	0.00%	6.65h
Gurobi	Exact	16.55	0.00%	45.63h	23.53	0.00%	48h
LKH-3 (default)	Heuristics	16.55	0.00%	46.28m	23.53	0.00%	2.57h
AM	RL+G	20.02	20.99%	**0.47s**	28.52	21.21%	1.09s
GCN	SL+G	29.72	79.61%	6.67m	43.15	83.38%	28.52m
POMO	RL+G	19.13	15.62%	0.49s	27.15	15.38%	**0.94s**
POMO(S)	RL+S	18.59	12.32%	2.97m	28.34	20.44%	3.25m
DIMES	RL+G	18.93	14.38%	0.97m	27.23	15.73%	2.08m
DIMES	RL+AS+G	17.81	7.61%	2.10h	25.11	6.72%	4.49h
LEHD	RL/SL+G	16.82	1.64%	1.82s	24.29	3.23%	3.84s
DIFUSCO	SL+G	18.11	9.41%	5.70m	25.68	9.14%	11.5m
T2T	SL+G	17.39	5.09%	4.90m	25.17	8.87%	15.66m
Prefix_Difusco	SL+G	17.92	8.23%	0.14m	25.86	9.91%	0.35m
Prefix_Difusco(16 samples)	SL+S	17.31	4.58%	2.10m	24.96	6.07%	5.94m
HyCO(POMO)	**RL+SL+G**	17.24	3.56%	1.20m	24.61	4.58%	2.43m
HyCO(LEHD)	**RL+SL+G**	**16.81**	**1.57%**	0.35m	**24.24**	**3.04%**	0.36m
POMO	RL+G+2OPT	17.77	7.37%	0.58s	25.10	6.67%	1.05s
DIMES	RL+G+2OPT	17.65	6.62%	1.01m	24.83	7.38%	2.29m
DIMES	RL+AS+G+2OPT	17.31	4.57%	2.10h	24.33	5.22%	4.49h
LEHD	RL/SL+G+2OPT	16.77	1.40%	1.83s	24.06	2.29%	3.85s
DIFUSCO	SL+G+2OPT	16.83	1.68%	5.75m	23.92	1.66%	17.52m
T2T	SL+G+2OPT	16.86	1.92%	2.42m	23.86	1.42%	15.90m
HyCO(POMO)	**RL+SL+G+2OPT**	16.82	1.63%	2.32m	23.92	1.66%	4.40m
HyCO(LEHD)	**RL+SL+G+2OPT**	**16.74**	**1.2%**	0.36m	**23.64**	**0.46%**	0.36m

B.2 MIS Performance Results

To validate the versatility of HyCO beyond routing tasks, we extended our evaluation to the MIS problem, a fundamental graph covering challenge. It is crucial to clarify the definition of the sampling budget S to appreciate the efficiency gains fully. For the baseline, S denotes the number of independent trajectories generated from scratch; for HyCO, S represents the number of top-ranked RL candidates selected at the trigger step for diffusion completion. As detailed in Table 7, we benchmarked HyCO against the specialized RL baseline (LWD) and the Diffusion backbone (prefix_COExpander) under an identical inference budget ($S = 5$). This setup allows us to rigorously assess whether the performance gains stem from our adaptive trigger and hybrid mechanism rather than simply increased computational resources.

The results demonstrate a strong synergy similar to our findings in TSP. HyCO consistently outperforms both learning-based baselines across all datasets, with the advantage being most pronounced on the challenging ER-700-800 benchmark. While the standalone RL policy suffers a significant 13.16% optimality gap and the pure Diffusion model plateaus at 9.16%, HyCO effectively combines their strengths to achieve a much lower gap of **5.76%**. Similarly, on RB-Large, HyCO reduces the RL error by nearly two-thirds (from 15.31% to 5.28%).

Notably, HyCO ($S = 5$) not only surpasses the baseline at the same budget but also remains competitive with or superior to baselines using significantly larger budgets (e.g., $S = 64$, as observed in typical diffusion evaluations). This indicates that the performance gain is attributable to the *quality* of the trigger-guided subspace exploration rather than the *quantity* of blind sampling, a phenomenon consistent with our observations in the TSP experiments.

Table 7: Performance comparison across RB-LARGE, ER-700-800, and SATLIB benchmarks for the Maximum Independent Set (MIS) problem. Obj. denotes the average node number of independent sets (higher is better). Gap indicates the percentage deviation from the optimal solution (lower is better). Time is the average inference time in seconds. S: Sample Decoding. S_{cand} is the candidate number of full DM proposals $\mathcal{P}_{DM}$.

METHOD	RB-LARGE			ER-700-800			SATLIB		
	Obj.↑	Gap↓	Time↓	Obj.↑	Gap↓	Time↓	Obj.↑	Gap↓	Time↓
KaMIS	43.00	0.00%	56.97s	44.97	0.00%	60.75s	425.95	0.00%	24.37s
Gurobi	42.19	1.83%	33.84s	38.78	13.75%	60.49s	425.92	0.01%	13.47s
LWD (RL)	36.67	15.31%	**0.85s**	39.05	13.16%	**0.64s**	421.80	1.04%	0.66s
prefix_Coexpander (Greedy)	40.05	6.84%	1.15s	40.18	10.65%	0.71s	421.84	0.96%	**0.25s**
prefix_Coexpander (s=5)	40.06	6.83%	1.22s	40.85	9.16%	0.96s	424.17	0.42%	0.91s
prefix_Coexpander(s=64)	40.58	5.63%	4.48s	40.91	9.03%	4.03s	424.78	0.27%	4.27s
HyCO (S_{cand}=5)	**40.73**	**5.28%**	2.69s	**42.38**	**5.76%**	4.69s	**425.40**	**0.12%**	6.31s

B.3 OP Performance Results

While our primary analysis focuses on TSP and MIS, we further evaluate HyCO on the Orienteering Problem (OP) to assess its generality on a different routing problem. We consider OP-50, OP-100, and OP-200, and compare HyCO against both the standalone RL backbone AM and the diffusion backbone prefix_Coexpander under different sampling budgets, as summarized in Table 8. Across all scales, HyCO consistently achieves the best performance among learning-based methods, improving over both the RL and diffusion baselines.

The advantage becomes more pronounced as the problem size increases. On OP-200, AM obtains an optimality gap of 13.05%, while prefix_Coexpander with the same sampling budget $s = 8$ reduces the gap to 4.15%. HyCO further lowers the gap to **1.08%**, corresponding to a 91.7% relative reduction over AM, and a 74.0% relative reduction over prefix_Coexpander $(s = 8)$. Notably, HyCO also outperforms the much stronger prefix_Coexpander $(s = 64)$ baseline, which attains a gap of **2.77%** on OP-200. This suggests that the improvement is not merely due to increasing the number of diffusion samples, but rather comes from using an informative RL prefix and an adaptive handover to guide diffusion generation toward a more favorable subspace.

HyCO achieves these gains with moderate computational overhead. For OP-200, HyCO takes **2.70s** per instance, which is comparable to prefix_Coexpander $(s = 64)$ and substantially faster than the Gurobi-300 reference. These results demonstrate that HyCO can serve as a plug-and-play hybrid inference mechanism beyond TSP, effectively combining weaker individual components into a stronger solver through adaptive RL-to-DM handover.

B.4 Adaptive Trigger vs. Fixed-time Triggering

To validate the effectiveness of our adaptive trigger, we compare HyCO against a fixed-time trigger baseline. Specifically, we sweep over fixed triggering steps k on a validation split and report the best-performing schedule (*Best Fixed*). This provides a strong reference for quantifying the benefit of adaptive triggering. The fixed-time baseline retains the same HyCO inference pipeline, and replaces only the adaptive trigger with a validation-selected fixed handover step.

Figure 2 reports the percentage optimality gap to the exact solver's solution for both TSP and MIS. Across all datasets, HyCO with adaptive triggering consistently outperforms the best fixed-time baseline. On TSP benchmarks, adaptive triggering yields improvements over a validation-selected schedule, reducing the gap from 0.026% to 0.020% on TSP-100 and from 5.66% to 3.56% on TSP-500.

Table 8: Performance comparison across OP-50, OP-100, and OP-200 benchmarks. Obj. denotes the average collected prize (higher is better). Gap indicates the percentage deviation from the optimal solution (lower is better). Time is the average inference time in seconds. S: Sample Decoding. S_{cand} is the candidate number of full DM proposals $\mathcal{P}_{DM}$.

Method	**OP-50**			**OP-100**			**OP-200**		
	Obj. ↑	Gap ↓	Time ↓	Obj. ↑	Gap ↓	Time ↓	Obj. ↑	Gap ↓	Time ↓
Gurobi-300	14.37	0.00%	300.0s	32.10	0.00%	300.0s	44.38	0.00%	300.0s
Gurobi-30	14.31	0.40%	30.0s	31.13	3.02%	30.0s	31.37	29.31%	30.0s
AM (RL)	11.92	17.04%	**0.03s**	28.37	11.62%	**0.03s**	38.59	13.05%	**0.09s**
prefix_Coexpander (Greedy)	12.27	14.61%	0.09s	28.64	10.77%	0.15s	41.36	6.80%	0.19s
prefix_Coexpander (s=8)	12.42	14.01%	0.12s	29.05	9.49%	0.21s	42.54	4.15%	0.47s
prefix_Coexpander (s=64)	12.51	12.94%	1.08s	29.20	9.03%	0.89s	43.15	2.77%	2.67s
HyCO (S_{cand}=8)	**13.23**	**7.93%**	0.55s	**29.51**	**8.06%**	0.41s	**43.90**	**1.08%**	2.70s

A similar trend holds on MIS benchmarks. On SATLIB, adaptive triggering reduces the gap from 1.20% to 0.12% (1.08% absolute improvement), bringing HyCO close to the exact reference. On ER-700–800, where instances vary significantly due to random graph generation, adaptive triggering still improves over the best fixed schedule by 3.22%. Overall, these results support that HyCO's trigger functions as a statistically effective triggering proxy: it reliably identifies a regime where conditional global completion provides net gains, without requiring any fixed schedule tuned per dataset or attempting to approximate the expectation-level optimal trigger step on each instance.

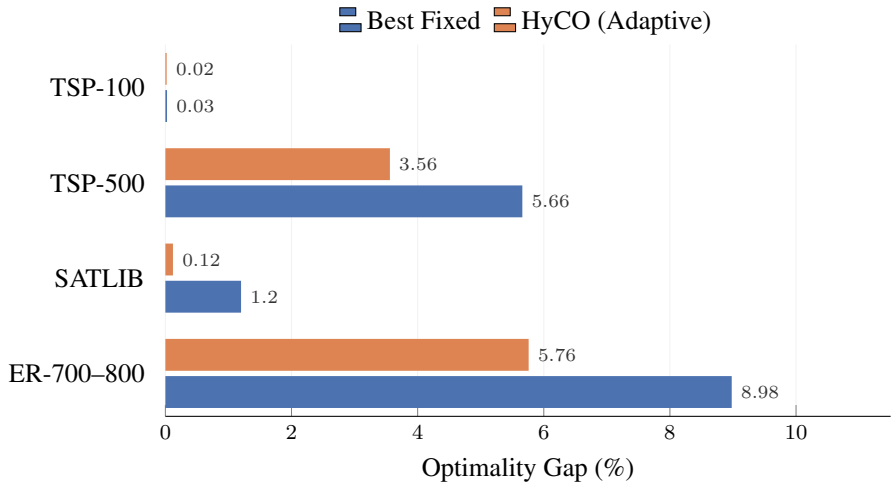


Figure 2: Adaptive trigger vs. best fixed-time trigger across tasks. Best Fixed is selected by validation sweeping over fixed triggering steps. We report the percentage optimality gap to the exact solver's solution for TSP and MIS. Lower is better.

B.5 Cross-Dataset Trigger Statistics

Here we report the average trigger step across different problem classes and scales in Table 9. Under the experimental settings used in this paper, the handover timing varies substantially across tasks and scales rather than following a fixed early-trigger pattern. While TSP-100 tends to trigger very early, larger TSP instances already exhibit later handovers. On MIS and OP benchmarks, the trigger can occur substantially later, suggesting that the RL solver can remain active for a nontrivial prefix before the conditional DM is invoked.

Table 9: Average trigger step across datasets. The handover timing varies substantially across problem classes and scales, indicating that HyCO uses an adaptive trigger rather than a fixed early handover.

	TSP			MIS			OP		
Scale / Benchmark	100	500	1000	RB-LARGE	ER-700–800	SATLIB	50	100	200
Avg. Trigger Step	0.52	6.00	8.11	56.14	10.09	81.77	3.46	1.79	20.06

C Ablation Study of HyCO

This section presents additional ablation studies on the robustness and design choices of HyCO. Section C.1 analyzes trigger-threshold calibration across TSP-500 and MIS ER-700–800, together with the sensitivity of other inference-time parameters on TSP-100. Section C.2 examines the contribution of combining policy entropy with RL–DM disagreement. Section C.3 evaluates the complementarity of the hybrid path and DM proposal tracks. All TSP ablation experiments are evaluated on 128 test instances, whereas the MIS threshold-sensitivity experiments use a subset of 50 test instances from ER-700–800.

C.1 Hyperparameter Sensitivity of Triggering and DM Exploration

To assess the robustness of HyCO's inference-time design, we analyze trigger thresholds across two settings and probe and exploration hyperparameters on TSP-100. We first evaluate threshold calibration on TSP-500 with LEHD and Prefix-Difusco, and on MIS ER-700–800 with LWD and Prefix-CoExpander. We then examine the sensitivity of the DM energy-probe timestep, the RL candidate-pool size for KL estimation, and the DM exploration budget on TSP-100.

For each setting, we sweep one trigger threshold while holding the other at its deployed value. Candidate thresholds are obtained by pooling the corresponding entropy or KL values over calibration trajectories and selecting empirical quantiles at the 25th, 50th, 75th, 90th, 95th, and 99th percentiles. We also include the deployed threshold where applicable, together with *always* and *never* controls. The *never* control disables only the swept trigger; the other trigger remains active. This scale-aware procedure covers low-, middle-, and upper-tail signal regimes without requiring arbitrary raw threshold values.

Table 10: Entropy-threshold H_{th} sensitivity across TSP-500 with LEHD and Prefix-Difusco and MIS ER-700–800 with LWD and Prefix-CoExpander. The KL threshold is fixed at $D_{\mathrm{th}} = 30$ for TSP-500 and $D_{\mathrm{th}} = 6.8155$ for MIS. Gap (%) is lower when better.

	TSP-500			MIS ER-700–800		
Percentile / control	**Threshold**	**Gap ↓**	**Avg. step**	**Threshold**	**Gap ↓**	**Avg. step**
Always	Always	1.653%	1.00	Always	9.98%	0.00
25th	0.0011	1.653%	1.00	0.1264	9.98%	0.00
50th	0.0734	1.673%	1.03	0.7713	9.85%	1.38
75th	0.5594	1.647%	1.42	1.0002	9.36%	23.24
90th	0.9480	1.600%	3.08	1.0772	**9.23%**	36.76
95th	1.1569	1.598%	6.04	1.0831	9.50%	40.06
99th	1.5233	**1.570%**	10.77	1.0886	10.56%	45.15
Never	Never	1.635%	2.33	Never	12.52%	53.35

The TSP-500 deployed entropy threshold is 1.2 with the gap 1.586%; the MIS deployed threshold is 0.8 with the gap 9.81%.

Across both problem families, the raw threshold values differ, but the quantile sweeps reveal a consistent calibration rule. Low- and mid-quantile thresholds tend to trigger diffusion prematurely, whereas effective configurations lie in a **high but non-extreme** signal region. On TSP-500, entropy thresholds from the 90th to the 99th percentile yield gaps between 1.570% and 1.600%; KL thresholds at the 95th percentile and above yield gaps between 1.586% and 1.622%. On MIS ER-700–800, the best results occur at the 90th percentile for entropy and the 95th percentile for KL. Raising

Table 11: KL-threshold D_{th} sensitivity across TSP-500 with LEHD and Prefix-Difusco and MIS ER-700–800 with LWD and Prefix-CoExpander. The entropy threshold is fixed at $H_{\text{th}} = 1.2$ for TSP-500 and $H_{\text{th}} = 0.8$ for MIS. Gap (%) is lower when better.

	TSP-500			MIS ER-700–800		
Percentile / control	**Threshold**	**Gap ↓**	**Avg. step**	**Threshold**	**Gap ↓**	**Avg. step**
Always	Always	1.653%	1.00	Always	9.98%	0.00
25th	0.3047	1.653%	1.00	4.0637	9.98%	0.00
50th	0.8069	1.690%	1.23	5.0974	9.98%	0.00
75th	1.3711	1.659%	2.24	5.4617	9.94%	0.12
90th	2.3090	1.663%	3.91	5.7318	9.76%	1.12
95th	3.4794	1.616%	4.06	5.9009	**9.58%**	1.68
99th	8.4543	1.622%	4.59	6.8155	9.81%	2.30
99.7th	30.0000	**1.586%**	5.87	–	–	–
Never	Never	1.590%	6.02	Never	9.81%	2.30

The TSP-500 deployed KL threshold is 30 with the gap 1.586%; the MIS deployed threshold is 6.8155 with the gap 9.81%.

the entropy threshold to the 99th percentile worsens the gap, suggesting that an overly restrictive threshold can delay useful DM invocation. These results support using empirical quantiles to calibrate setting-specific thresholds rather than transferring raw threshold values unchanged across problem families and backbone pairs.

Beyond threshold calibration, we study other inference-time parameters on TSP-100. These experiments examine the sensitivity of HyCO to the DM energy-probe timestep, the RL candidate-pool size used for KL estimation, and the DM exploration budget. They characterize how these choices affect solution quality and inference cost.

Table 12 examines the sensitivity of our framework to the DM energy probe's timestep (`dm_probe_timestep`). The results demonstrate remarkable robustness: across a wide range of timesteps from 100 to 900, the optimality gap remains stable at 0.02%. This is a significant practical advantage, as it indicates that the single-step energy probe is a reliable signal that does not require meticulous hyperparameter tuning.

Table 12: Sensitivity analysis on the DM energy probe timestep, `dm_probe_timestep`.

Energy Probe Ts	**Gap (%) ↓**	**Time (s)**	**Ave. trigger step (Trig. Rates)**
100	0.02	1.762s	0.66 (100.0%)
300	0.02	1.768s	0.66 (100.0%)
500	0.02	**1.749s**	0.52 (100.0%)
700	0.02	1.755s	0.51 (100.0%)
900	0.02	1.753s	0.51 (100.0%)

Table 13 investigates the impact of the RL candidate pool size (`probe_rl_top_m`) used for the KL divergence calculation. This experiment highlights the importance of providing a sufficiently large set of candidate actions for the probe. With too few candidates (e.g., 5), the KL divergence is not a reliable indicator, resulting in a poor optimality gap (1.30%) and a low trigger rate (43.0%). Our default setting of 15 ensures that the divergence metric is calculated over a meaningful distribution, allowing for effective detection of critical junctures.

Table 14 analyzes the effect of the candidate selection threshold for DM exploration (`TopN_cum_Th`). This parameter controls the breadth of the DM's search once it is triggered. The results show a clear trade-off: a small threshold (e.g., 0.2) is faster but often fails to find a high-quality solution, yielding a suboptimal 0.12% gap. Increasing the exploration breadth is crucial for capitalizing on the DM's generative power. Our default value of 0.8 allows the DM to explore a diverse set of high-probability candidates, which is vital for discovering the near-optimal paths that lead to our state-of-the-art 0.02% gap.

Table 13: Sensitivity analysis on the RL candidate pool size, `probe_rl_top_m`, used for KL divergence calculation.

`Probe_RL_Top_M`	**Gap (%) ↓**	**Time (s)**	**Ave. trigger step (Trig. Rates)**
5	1.30	0.761s	4.05 (43.0%)
10	0.15	1.278s	5.20 (97.1%)
15	0.02	1.749s	0.52 (100.0%)
20	0.01	2.364s	0.00 (100.0%)

Table 14: Ablation study on the candidate selection strategy for DM exploration, `n_cumulative_threshold`.

`TopN_cum_Th`	**Gap (%) ↓**	**Time (s)**	**Ave. trigger step (Trig. Rates)**
0.2	0.12	0.158s	0.52 (100.0%)
0.4	0.12	0.570s	0.52 (100.0%)
0.5	0.08	0.723s	0.52 (100.0%)
0.6	0.05	0.942s	0.52 (100.0%)
0.8	0.02	1.749s	0.52 (100.0%)

C.2 Necessity of Dual-Signal Trigger Design

Table 15 provides a comprehensive analysis of our core trigger mechanism. The results clearly show that the dual-criteria trigger, which combines both RL policy entropy and KL divergence, is essential for top performance. Relying solely on the KL-divergence or entropy trigger leads to significantly worse optimality gaps (0.09% and 0.05%, respectively). This confirms our hypothesis that RL policy uncertainty (entropy) and model disagreement (KL) are complementary signals. The former identifies when the RL agent is indecisive, while the latter detects when it is confidently wrong. Together, they form a robust and effective condition for invoking the diffusion model.

Table 15: Complete Ablation study on the core components of HyCO, evaluated on the TSP-100 dataset. Gap (%) is the optimality gap compared to the Concorde solver. Time (s) is the average inference time per instance. Avg. Trigger Step Index indicates the average step number in the tour construction at which the Diffusion Model was first invoked. Trigger Rate denotes the percentage of instances where the DM was triggered at least once. The best-performing model is highlighted in bold.

Model / Method	**Description: Triggers on**	**Gap (%) ↓**	**Time (s)**	**Ave. trigger step (Trig. Rates)**
Trigger Mechanism Ablation				
HyCO (Entropy)	policy entropy only	0.05	1.903s	2.36 (99.8%)
HyCO (KL)	KL divergence only	0.09	1.305s	1.35 (97.0%)
HyCO (Full Model)	**Full Model (Entropy + KL)**	**0.02**	1.749s	0.52 (100.0%)

C.3 Complementarity of the Two Output Tracks

To isolate the effect of output-track selection, we evaluate both tracks on the same 128 TSP-500 instances while keeping the adaptive trigger and DM candidate pool fixed. We change only which output HyCO returns. "Hybrid path only" returns the corrected RL continuation; "DM proposal only" returns the best conditional-DM proposal; and full HyCO returns the better solution from the two tracks. Table 16 reports the optimality gaps and the track selected by full HyCO for each instance.

We also counted which track supplied the solution selected by full HyCO on each instance. This reveals a clear backbone-dependent pattern: with POMO, the DM proposal track supplies the selected solution on 127 of 128 instances, whereas with LEHD, the hybrid path does so on 124 of 128 instances.

Table 16: Ablation of the two output tracks on TSP-500. Gap (%) is the optimality gap to the Concorde solver; lower is better. Track-selection counts are out of 128 instances.

Backbone pair	**Hybrid path only** **Gap (%)** $\downarrow$	**DM proposal only** **Gap (%)** $\downarrow$	**Full HyCO** **Gap (%)** $\downarrow$
POMO + Prefix-Difusco	13.78%	3.65%	**3.56%**
LEHD + Prefix-Difusco	1.61%	3.23%	**1.57%**

These results show that **the two tracks are complementary rather than uniformly beneficial**. With the weaker POMO backbone, the DM proposal track is selected on 127/128 instances and achieves a 3.65% gap, while the hybrid path alone has a 13.78% gap. With the stronger LEHD backbone, the pattern reverses: the hybrid path is selected on 124/128 instances and achieves a 1.61% gap, whereas the DM proposal track alone has a 3.23% gap. Full HyCO obtains the best result in both settings (3.56% and 1.57%, respectively). Thus, the dual-track design provides robustness to the relative strengths of the RL prefix and conditional DM completion, rather than relying on one track to dominate in all settings.

C.4 Effect of Multiple Triggering on Solution Quality and Inference Time

The single-handover design is motivated by the theoretical result, which identifies a one-switch optimum under the complementary error-profile assumptions. Because learned backbones may depart from these idealized profiles, we additionally evaluated an iterative multi-trigger variant using POMO as the RL backbone and Prefix-Difusco as the DM backbone. Whenever the trigger is activated again after RL continuation, the multi-trigger variant re-runs the full dual-track procedure. All data are shown in Table 17.

Table 17: Single-trigger and multiple-trigger HyCO with POMO and Prefix-Difusco. Gap (%) is the optimality gap to Concorde; lower is better. Time is the average inference time per instance.

Dataset	**Single trigger**		**Multiple triggers**	
	Gap $\downarrow$	**Time**	**Gap** $\downarrow$	**Time**
TSP-50	0.03%	0.25 s	0.02%	2.38 s
TSP-100	0.02%	1.20 s	0.01%	23.28 s
TSP-500	3.56%	1.20 min	1.37%	6.43 min
TSP-1000	4.58%	2.43 min	4.47%	14.22 min

The results show that multiple triggering consistently improves solution quality, but the gain is modest in most settings relative to its computational cost. On TSP-50, TSP-100, and TSP-1000, it improves the gap by only 0.01%, 0.01%, and 0.11%, respectively, while increasing inference time by 9.5$\times$, 19.4$\times$, and 5.9$\times$. On TSP-500, it yields a larger improvement, reducing the gap from 3.56% to 1.37%, but requires 5.4$\times$ more inference time, increasing from 1.20 to 6.43 minutes. Accordingly, the single-handover configuration is our default efficiency-oriented setting.

D Implementation Details of Prefix_Difusco and RL models

Here we provide detailed specifications for our backbone conditional diffusion model, Prefix_Difusco, used in the experiments. About the RL model, we uniformly use the code and environment provided by the RL4CO [39] package for training [3]. The detailed hyperparameters are listed in D.4.

D.1 Model Architecture

The core of Prefix_Difusco modifies the GNN architecture from DIFUSCO[8] for the conditional generation task. The key components are:

- **Node Feature Embedding**: Node coordinates are first converted into high-dimensional features using a sinusoidal positional embedding. A binary feature indicating whether a node is part of the prefix is concatenated to this embedding. A final linear layer projects this combined feature vector to the GNN's expected input dimension ($d_{node} = 128$).
- **PrefixEncoder**: An LSTM-based encoder takes the sequence of node features corresponding to the prefix tour and outputs a single global conditioning vector ($d_{cond} = 256$). This vector summarizes the properties of the given partial tour.
- **DifuscoGNNEncoder**: This is the main denoising network. It is a 12-layer GNN that processes a graph where nodes have the features described above. At each layer, the GNN's message passing is conditioned by both the global prefix vector from the `Prefix_Encoder` and a sinusoidal embedding of the current timestep t.
- **Output Head**: The GNN outputs a logit for each potential edge in the graph, representing the probability of that edge being part of the optimal tour.

D.2 Training Procedure

The model is trained to predict the ground-truth adjacency matrix x_0 from a noised version x_t and a conditional prefix.

- **Dataset and Conditioning**: The training dataset consists of TSP instances and their optimal tours. For each sample, we derive a training instance by randomly selecting a prefix of length k from the optimal tour.
- **Diffusion Process**: We use a discrete diffusion process over $T = 1000$ steps with a cosine noise schedule to corrupt the ground-truth adjacency matrix x_0 into a noisy matrix x_t.
- **Masked Loss Function**: The model's objective is to minimize the Binary Cross-Entropy (BCE) loss between the predicted adjacency matrix and the ground truth x_0. Crucially, the loss is only computed on the "suffix" edges—that is, all edges except those whose both endpoints are within the given prefix. This forces the model to learn how to best complete the tour.
- **Curriculum Learning**: To improve convergence and performance, we employ a multi-stage curriculum. The training starts with a distribution of long prefixes (e.g., $k \in [60, 90]$), making the completion task easier. As training progresses, the distribution of k shifts towards shorter, more difficult prefixes (e.g., $k \in [1, 30]$), allowing the model to gradually master the full conditional generation task.

D.2.1 Training for TSP-50

The `Prefix_Difusco` model for TSP-50 was trained from scratch. We employed a 5-stage curriculum learning strategy designed to gradually increase the task difficulty. Each stage was trained for 10 epochs, initializing from the best checkpoint of the previous stage.

- **Stage 1 (Easy):** Trained on long prefixes with lengths $k \in [30, 49]$.
- **Stage 2 (Medium):** Trained on prefix lengths $k \in [10, 30]$.
- **Stage 3 (Hard):** Trained on the full range of prefix lengths $k \in [1, 49]$.

[3] https://github.com/ai4co/rl4co

- **Stage 4 (Short Focus):** Focused on short prefixes with lengths $k \in [1, 20]$.
- **Stage 5 (Very Short Focus):** Further focused on very short prefixes with $k \in [1, 10]$ to enhance performance on early-step decisions.

D.2.2 Training for TSP-100

The model for TSP-100 was also trained from scratch following a similar multi-stage curriculum learning approach. The prefix length ranges for each stage were adjusted proportionally for the larger problem size to ensure a smooth learning progression from easy to hard completion tasks. The core hyperparameters, such as hidden dimensions and learning rate, were kept consistent with the TSP-50 model.

- **Stage 1 (Easy):** Trained on long prefixes with lengths $k \in [61, 99]$.
- **Stage 2 (Medium):** Trained on prefix lengths $k \in [30, 60]$.
- **Stage 3 (Hard):** Trained on the full range of prefix lengths $k \in [1, 99]$.
- **Stage 4 (Short Focus):** Focused on short prefixes with lengths $k \in [1, 20]$.
- **Stage 5 (Very Short Focus):** Further focused on very short prefixes with $k \in [1, 10]$ to enhance performance on early-step decisions.

D.2.3 Training for TSP-500

Due to the significantly larger scale of TSP-500, we adopted a more advanced training strategy combining **transfer learning** and a tailored curriculum.

- **Transfer Learning**: The TSP-500 model was not trained from scratch. Instead, it was initialized using the weights from our best-trained TSP-100 checkpoint. Weights were transferred for all layers with matching names and shapes (e.g., GNN layers, prefix encoder), providing a strong starting point and accelerating convergence.
- **Curriculum Learning**: After initialization, the model was fine-tuned on TSP-500 data using a 5-stage curriculum similar to the one for TSP-50, but with ranges adjusted for $N = 500$ (e.g., Stage 1: $k \in [50, 100]$, Stage 2: $k \in [20, 50]$, etc.). For the results reported in this paper, we ran an accelerated training schedule of approximately 20 epochs in total, focusing on the most critical curriculum stages to balance performance and computational cost.

D.2.4 Training for TSP-1000

For the largest scale, TSP-1000, we continued the strategy of combining transfer learning with a specialized curriculum to manage the increased complexity and computational demands.

- **Transfer Learning**: The TSP-1000 model was initialized with the weights from our best-performing TSP-100 checkpoint. This transfer learning approach provided a robust feature foundation, significantly accelerating the training convergence on the larger graph size.
- **Curriculum Learning**: Following initialization, the model was fine-tuned on the TSP-1000 dataset using a 4-stage curriculum over a total of 25 epochs. The training began with easier tasks (completing tours from long prefixes, with k up to 500) and progressively moved to more difficult scenarios, focusing on shorter prefixes (k down to 1) in later stages to refine the model's ability to make critical early decisions.

D.3 Decoding Algorithms

To convert the probabilistic heatmap output from our `Prefix_Difusco` model into a valid TSP tour, we employ a deterministic greedy decoding strategy inspired by DIFUSCO [8]. This approach ensures that given a heatmap, the resulting tour is always the same, which is crucial for the stability of the HyCO framework. The core decoding process follows a principled, multi-stage procedure designed to construct high-quality tours while respecting the prefix constraints.

The decoding algorithm proceeds as follows:

- **Edge Score Calculation:** The raw adjacency probability matrix P from the diffusion model is first symmetrized to ensure consistency ($P' = (P+P^T)/2$). To favor shorter edges, which are fundamental to good TSP solutions, we compute an edge score for each potential edge (i, j) by dividing its symmetrized probability by its Euclidean distance: $S_{ij} = P'_{ij}/\text{dist}(i, j)$. All possible edges are then sorted in descending order based on these scores.
- **Enforce Prefix Constraint:** Before any greedy selection, the decoder first enforces the given conditional prefix. All edges that form the given partial tour are mandatorily included in the solution set. A Union-Find data structure is initialized, and the degrees of the prefix nodes are updated accordingly to ensure these edges are fixed.
- **Greedy Spanning Path Construction:** The algorithm iterates through the globally sorted list of edges. For each candidate edge, it performs three checks:
  1. It is not an existing prefix edge.
  2. Adding the edge will not result in any node having a degree greater than two.
  3. Adding the edge will not form a premature cycle (verified using the Union-Find data structure).

  If all conditions are met, the edge is added to the solution set, and the node degrees and Union-Find structure are updated. This process continues until a total of $N - 1$ edges have been selected, forming a spanning path of all nodes.
- **Tour Finalization:** Once a spanning path of $N - 1$ edges is formed, there will be exactly two nodes with a degree of one (the endpoints of the path). The final edge connecting these two endpoints is deterministically added to close the path and form a valid Hamiltonian cycle. The final list of N edges is then converted into a sequential tour starting from the first node of the original prefix (or node 0 if no prefix was given).

D.4 Hyperparameters

This subsection provides a comprehensive summary of the hyperparameters for the core models employed in the HyCO framework: The configurations for our conditional diffusion model (`Prefix_Difusco`), the Attention Model (AM), and POMO [5, 16] are detailed in Table 18, Table 19, and Table 20, respectively. To ensure rigorous comparability, we strictly adopted the architectural configurations (e.g., layers, dimensions) from the original DIFUSCO and RL4CO implementations, thereby isolating the performance gains to the HyCO framework logic rather than backbone scaling. Besides, for inference-specific parameters such as DDIM sampling steps, values were selected based on a preliminary analysis of the speed-quality trade-off.

E Limitations and Broader Impacts

HyCO's theoretical analysis relies on explicit error-scaling and boundary assumptions on the RL and DM backbones. Extending the framework to other solver combinations may require re-examining these assumptions and adapting the trigger signals accordingly.

HyCO is a general neural combinatorial optimization method and is not tied to a specific high-stakes deployment. Its potential positive impact lies in improving the efficiency and quality of optimization systems, with possible applications in routing, scheduling, logistics, and resource allocation. Potential negative impacts could arise if such tools are deployed in sensitive domains where suboptimal objectives or incorrect solutions affect access to resources or services. In such settings, deployment should include domain-specific constraints, validation, and human oversight.

Table 18: Hyperparameters for Prefix_Difusco across different problem sizes.

Parameter	TSP-50	TSP-100	TSP-500	TSP-1000
Model Architecture				
Node Count (N)	50	100	500	1000
sparse factor(K)	N.A	N.A	N.A	100
GNN Layers (L)	12	12	12	12
Hidden Dimension	256	256	256	256
Node Embedding Dim	128	128	128	128
Prefix Condition Dim	256	256	256	256
Diffusion Process				
Timesteps (T)	1000	1000	1000	1000
Beta Schedule	cosine	cosine	cosine	cosine
Inference Steps	10	10	50	50
Inference Sampler	DDIM	DDIM	DDIM	DDIM
Training				
Batch Size (per GPU)	128	96	4	8
Epoch	50	50	20	25
Training data (per epoch)	1500000	1500000	128000	65000
Learning Rate	2e-4	2e-4	2e-5	1e-4
Optimizer	Adam	Adam	Adam	Adam
Training Method	Curriculum	Curriculum	Transfer&Curriculum	Transfer&Curriculum
Environment	2x NVIDIA A40 GPUs			

Table 19: Hyperparameters for Attention Model across different problem sizes. Need to notice, AM and POMO models for TSP-500 are trained on the tsp-200 instances and then generalized into the TSP 500.

Parameter	TSP-50	TSP-100	TSP-500
Model Architecture			
GNN Layers (L)	3	3	3
Hidden Dimension	512	512	512
Node Embedding Dim	128	128	128
Attention heads	8	8	8
Baseline	rollout	rollout	critic
Training			
Batch Size (per GPU)	512	512	1024
Training data (each epoch)	1280000	1280000	1280000
Epoch	100	100	120
Learning Rate	1e-4	1e-4	1e-4
LR Scheduler	multistep LR, gamma=0.1, milestone = [80, 95]		
Normalization	batch		
Environment	1x NVIDIA A40 GPUs		

Table 20: Hyperparameters for POMO across different problem sizes.

Parameter	TSP-50	TSP-100	TSP-500
Model Architecture			
GNN Layers (L)	6	6	6
Hidden Dimension	512	512	512
Node Embedding Dim	128	128	128
Attention heads	8	8	8
Augment	8	8	8
Training			
Batch Size (per GPU)	512	512	256
Training data (each epoch)	100000	100000	100000
Epoch	400	800	400
Learning Rate	1e-4	1e-4	1e-4
LR Scheduler	multistep LR, gamma=0.1, milestone = [80, 95]		
Normalization	instance		
Environment	1x NVIDIA A40 GPUs		2x NVIDIA A40 GPUs

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 state claims within the paper's scope: the theory is qualified by explicit assumptions, the algorithm is framed as RL–DM hybrid inference, and the empirical claims are supported by TSP, MIS, OP, oracle-validation, and ablation results.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We provide this in the Appendix.

   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: All assumptions and proofs are provided in the main text or appendix.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We submit the codes with the paper, and hyperparameters are also included in the Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often

one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: The code is provided in the supplemental material.

   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: All details and hyperparameters are included in the Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: The paper reports averaged benchmark performance together with sensitivity analyses and ablations supporting robustness, but it does not provide systematic error bars or statistical significance tests for all main results.

   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: All are provided in the paper.

   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 follow all the NeurIPS Code of Ethics.

Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss this in Appendix E.

    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: [No]

    Justification: The paper does not release high-risk models, scraped datasets, or assets with substantial misuse risk; the work focuses on neural combinatorial optimization benchmarks.

    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: Existing solvers, benchmark datasets, pretrained checkpoints, and baseline methods are credited through citations; we use standard public CO benchmarks and official pretrained checkpoints where applicable.

    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: [N/A]

    Justification: The paper does not introduce or release a new dataset or high-risk asset.

    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 work does not involve crowdsourcing, user studies, or 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 work does not involve human subjects, so IRB approval is not applicable.

    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: LLMs are not used as an important, original, or non-standard component of the proposed method or experiments.

    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.