IN-CONTEXT LEARNING FOR LATENT-SPACE BAYESIAN OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent-space Bayesian optimization (LSBO) enables sample-efficient optimization over structured objects such as molecules by searching in the continuous space of a generative model. However, its surrogate is typically fitted anew for each objective from limited observations, without transferring experience across tasks. We investigate whether pretrained in-context predictors can provide such transfer in LSBO. Using prior-data fitted networks (PFNs), we introduce PILBO (*Pre-trained In-context Latent-space Bayesian Optimization*), a framework combining pretrained in-context prediction with latent-space molecular generation. Its main implementation predicts from molecular representations independent of the generator and achieves the strongest aggregate performance on GuacaMol optimization objectives. We further introduce a continued-pretraining (CPT) scheme based on handcrafted synthetic molecular objectives and BO-oriented episode construction. The pretrained PFN already captures most of the benefit, with CPT yielding only modest further gains. Our analyses suggest that this limited transfer reflects a mismatch between the offline synthetic episodes used for adaptation and the adaptive prediction problems encountered during BO. Overall, pretrained in-context surrogates provide a strong alternative to repeatedly fitted LSBO surrogates.

1 INTRODUCTION

Many scientific design problems require optimizing expensive objectives over large, structured spaces, with molecular design, protein engineering, and biological sequence optimization being prominent examples (Gomez-Bombarelli et al., 2018; Gruver et al., 2024; Torres et al., 2026). Bayesian optimization (BO) is well suited to such settings because it uses a probabilistic surrogate to guide evaluations under a limited budget (Garnett, 2023). For discrete objects such as molecules, latent-space Bayesian optimization (LSBO) makes this search tractable by optimizing in the continuous space of a generative model and decoding selected points back to valid candidates (Gomez-Bombarelli et al., 2018). This paradigm has enabled increasingly effective molecular optimization under small evaluation budgets and provides a general route for applying BO to structured design spaces.

LSBO typically uses a variational autoencoder (VAE) to embed structured objects in a continuous search space. The resulting latent representation must then serve both generation and prediction, although a geometry learned for reconstruction need not align with the downstream objective and may change as the VAE is updated. COWBOYS avoids this coupling by predicting from molecular structure while retaining the VAE only for candidate generation (Moss et al., 2025). Yet the surrogate itself remains learned separately for every optimization problem: a Gaussian process (GP) starts each new objective from the observations collected in that run alone. Under the small budgets typical of BO, early posterior or hyperparameter errors can mis-rank candidates, distort subsequent data collection, and propagate through the optimization trajectory. Moreover, no statistical experience transfers across previously solved molecular optimization problems.

In-context learning offers a way to amortize prediction across tasks: a pretrained model adapts to a new problem by conditioning on observed examples rather than updating its parameters. In tabular domains, this idea has led to foundation models pretrained at scale over massive collections of real or synthetic tabular tasks, enabling strong regression and classification from small datasets (Zhao et al., 2025; Hollmann et al., 2025; Qu et al., 2026; Ma et al., 2026). Prior-data fitted networks (PFNs) are a prominent instance, pretrained on large distributions of synthetic tasks to approximate Bayesian prediction in context (Muller et al., 2021; Grinsztajn et al., 2026). Their recent use as BO surrogates

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

(Muller et al., 2023; Yu et al., 2026) makes them natural candidates for LSBO, replacing per-task Gaussian-process fitting with transferable in-context prediction.

Applying pretrained PFNs to LSBO raises two linked questions: which surrogate representation best supports transfer, and whether domain-specific adaptation can improve it further. These questions remain largely unexplored in LSBO, despite the growing use of PFNs in conventional BO.

Contributions. We introduce PILBO, a unified in-context LSBO framework based on PFNs, and study how surrogate representation and offline adaptation affect molecular optimization. Our contributions are:

- **Methodological.** We develop a common in-context surrogate formulation for LSBO that allows molecular prediction to be decoupled from latent-space generation and includes direct prediction in VAE coordinates as a special case. We further design a molecular continued-pretraining (CPT) distribution with optimization-biased sampling and embedding-guided task curation.
- **Empirical.** We benchmark PILBO against established LSBO baselines. Its decoupled configuration achieves the strongest aggregate performance, while the latent-input configuration remains competitive. The original pretrained PFN already accounts for most of the gain, with CPT providing modest additional benefit.
- **Analytical.** We use ablations and deployment diagnostics to study which aspects of representation and continued pretraining matter for in-context LSBO. Our analyses reveal substantial mismatch between offline molecular adaptation and the adaptive prediction problems encountered during BO, helping explain why domain-specific CPT provides only limited additional gains.

2 Background

2.1 Latent-space Bayesian optimization

Let $\mathcal{X}$ be a structured design space and $f : \mathcal{X} \to \mathbb{R}$ an expensive objective. We seek to solve

$$\boldsymbol{x}^\star \in \arg\max_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}). \tag{1}$$

A VAE defines a latent prior $p_0(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{0}, \boldsymbol{I}_d)$, an amortized variational posterior $q_{\boldsymbol{\psi}}(\boldsymbol{z} \mid \boldsymbol{x})$, and a decoder $p_{\boldsymbol{\psi}}(\boldsymbol{x} \mid \boldsymbol{z})$, with $\mathcal{Z} \subseteq \mathbb{R}^d$. We denote by

$$e_{\boldsymbol{\psi}}(\boldsymbol{x}) = \mathbb{E}_{q_{\boldsymbol{\psi}}(\boldsymbol{z}|\boldsymbol{x})}[\boldsymbol{z}] \tag{2}$$

the posterior-mean encoder used to assign latent coordinates to observed objects. Thus, an evaluated object $\boldsymbol{x}_i$ is associated with $\boldsymbol{z}_i = e_{\boldsymbol{\psi}}(\boldsymbol{x}_i)$. After t evaluations, we write the observed history as

$$\mathcal{D}_t = \{(\boldsymbol{x}_i, \boldsymbol{z}_i, y_i)\}_{i=1}^t, \qquad \boldsymbol{z}_i = e_{\boldsymbol{\psi}}(\boldsymbol{x}_i), \qquad y_i = f(\boldsymbol{x}_i) + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2). \tag{3}$$

The decoder induces an objective over the latent space,

$$g_{\boldsymbol{\psi}}(\boldsymbol{z}) = \mathbb{E}_{\boldsymbol{x}\sim p_{\boldsymbol{\psi}}(\cdot|\boldsymbol{z})}\left[f(\boldsymbol{x})\right], \tag{4}$$

so that LSBO may instead search for

$$\boldsymbol{z}^\star = \arg\max_{\boldsymbol{z}\in\mathcal{Z}} g_{\boldsymbol{\psi}}(\boldsymbol{z}). \tag{5}$$

For a deterministic decoder $h_{\boldsymbol{\psi}}$, this reduces to $g_{\boldsymbol{\psi}} = f \circ h_{\boldsymbol{\psi}}$.

Standard LSBO fits a probabilistic surrogate for the relationship between $\boldsymbol{z}$ and y using the observations in $\mathcal{D}_t$, and selects the next latent query through an acquisition function α_t:

$$\boldsymbol{z}_{t+1} = \arg\max_{\boldsymbol{z}\in\mathcal{Z}} \alpha_t(\boldsymbol{z}), \qquad \boldsymbol{x}_{t+1} \sim p_{\boldsymbol{\psi}}(\cdot \mid \boldsymbol{z}_{t+1}). \tag{6}$$

For the deterministic decoder used in our experiments, the second relation reduces to $\boldsymbol{x}_{t+1} = h_{\boldsymbol{\psi}}(\boldsymbol{z}_{t+1})$. The surrogate is commonly a GP, with acquisition strategies such as expected improvement or Thompson sampling (Rasmussen & Williams, 2006; Garnett, 2023). In practice, LSBO methods often restrict the search to local regions and periodically update the VAE using observations collected during optimization (Maus et al., 2022).

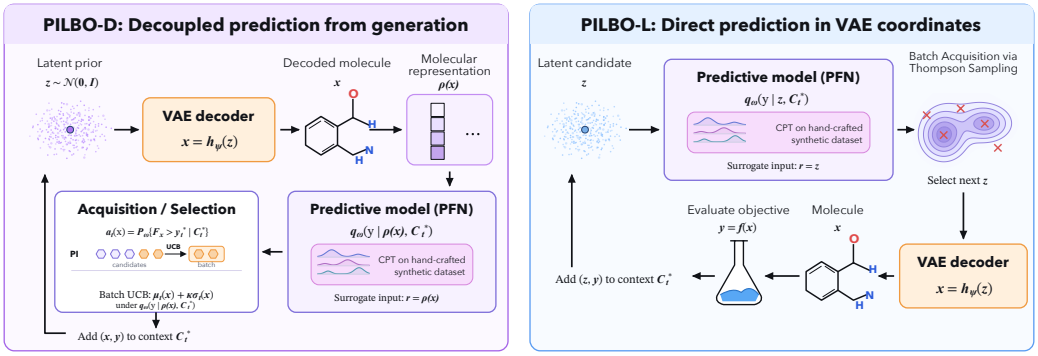

Figure 1: **Overview of PILBO.** The main decoupled configuration, PILBO-D, predicts from molecular representations while using latent coordinates for generation (left). The latent-input specialization, PILBO-L, uses VAE coordinates for both prediction and generation (right).

2.2 Prior-data fitted networks

A PFN is a predictor trained across a distribution of datasets rather than fitted separately to each new dataset (Muller et al., 2021). A regression task can be seen as generated from latent task parameters η,

$$\eta \sim p(\eta), \qquad y_i \sim p(y_i \mid \boldsymbol{u}_i, \eta). \tag{7}$$

Given a context $\mathcal{C} = \{(\boldsymbol{u}_i, y_i)\}_{i=1}^{n_{\text{ctx}}}$ and a query $\boldsymbol{u} \in \mathbb{R}^p$, Bayesian prediction marginalizes these latent parameters,

$$p(y \mid \boldsymbol{u}, \mathcal{C}) = \int p(y \mid \boldsymbol{u}, \eta)\, p(\eta \mid \mathcal{C})\, d\eta. \tag{8}$$

PFNs amortize this inference procedure with a neural predictor $q_{\boldsymbol{\omega}}(y \mid \boldsymbol{u}, \mathcal{C})$. This yields an in-context learning mechanism: adaptation to a new regression task occurs through conditioning on $\mathcal{C}$, not via parameter updates. In pretraining, a dataset sampled from a task distribution $\mathcal{P}$ is split into a context set $\mathcal{C} = \{(\boldsymbol{u}_i, y_i)\}_{i=1}^{n_{\text{ctx}}}$ and a query set $\mathcal{Q} = \{(\boldsymbol{u}_j, y_j)\}_{j=1}^{n_{\text{qry}}}$. The model is trained by minimizing

$$\mathcal{L}_{\text{PFN}}(\boldsymbol{\omega}) = \mathbb{E}_{(\mathcal{C},\mathcal{Q})\sim\mathcal{P}} \left[-\frac{1}{|\mathcal{Q}|} \sum_{(\boldsymbol{u},y)\in\mathcal{Q}} \log q_{\boldsymbol{\omega}}(y \mid \boldsymbol{u}, \mathcal{C}) \right]. \tag{9}$$

3 In-context latent-space Bayesian optimization

PILBO combines a pretrained in-context surrogate with latent-space molecular generation. Section 3.1 introduces the common predictive formulation, independently of the representation supplied to the surrogate. Section 3.2 develops our main decoupled implementation, PILBO-D, and Section 3.3 describes molecular continued pretraining. Section 3.4 then specializes the surrogate formulation to direct prediction in VAE coordinates, giving PILBO-L.

3.1 In-context surrogate

Let $r : \mathcal{X} \to \mathbb{R}^{d_r}$ denote the representation assigned to observed molecules. It induces the context and predictive model

$$\mathcal{C}_t^r = \{(r(\boldsymbol{x}_i), y_i)\}_{i=1}^{t}, \qquad q_{\boldsymbol{\omega}}(y \mid \boldsymbol{u}, \mathcal{C}_t^r), \quad \boldsymbol{u} \in \mathbb{R}^{d_r}. \tag{10}$$

The map r may use VAE coordinates (direct prediction) or molecular descriptors (decoupled prediction). In either case, the current context determines the predictive distribution, from which we obtain the quantities required by the acquisition rule. We initialize the predictor $q_{\boldsymbol{\omega}}(y \mid \boldsymbol{u}, \mathcal{C})$ from an official TabPFN-3 checkpoint (Grinsztajn et al., 2026), and optionally continue pretraining it on molecular tasks (see Section 3.3). Its weights remain fixed throughout each BO run. The representation and the candidate-generation procedure determine how a proposed design is presented as a query vector $\boldsymbol{u}$.

3.2 Decoupling prediction from generation

Following the success of COWBOYS (Moss et al., 2025), PILBO-D decouples prediction from generation by using a molecular representation independent of the VAE latent coordinates. Concretely,

we set

$$r(\boldsymbol{x}) = \rho(\boldsymbol{x}), \qquad \rho : \mathcal{X} \to \mathbb{R}^{d_\rho}, \tag{11}$$

where ρ is a fixed representation of the decoded molecule, such as RDKit2D descriptors. The VAE is then used only to generate candidate molecules, while the PFN predicts their objective values from $\rho(\boldsymbol{x})$ through the common in-context interface (Section 3.1). Unlike VAE coordinates, this representation remains tied to the molecule that is ultimately evaluated and is unaffected by changes in the generative latent space.

Conditional generation. COWBOYS (Moss et al., 2025) biases molecular generation toward promising candidates by defining an improvement-weighted distribution over the VAE latent space and sampling from it with MCMC. We follow this conditional-generation principle, replacing the GP surrogate with the PFN predictive distribution.

For a decoded molecule $\boldsymbol{x}$, let

$$F_{\boldsymbol{x}} \mid \mathcal{C}_t^\rho \sim q_{\boldsymbol{\omega}}(\cdot \mid \rho(\boldsymbol{x}), \mathcal{C}_t^\rho)\,, \tag{12}$$

and let $y_t^\star = \max_{i \le t} y_i$ denote the incumbent. We define its probability of improvement as

$$a_t(\boldsymbol{x}) = \mathbb{P}_{\boldsymbol{\omega}}\left(F_{\boldsymbol{x}} > y_t^\star \mid \mathcal{C}_t^\rho\right). \tag{13}$$

For the deterministic decoder $h_{\boldsymbol{\psi}}$, this implicitly induces the latent-space target

$$\pi_t(\boldsymbol{z}) \propto a_t(h_{\boldsymbol{\psi}}(\boldsymbol{z}))\, p_0(\boldsymbol{z}), \tag{14}$$

where $p_0(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \mathbf{0}, \boldsymbol{I}_d)$ is the VAE prior. The latent variable therefore serves only as a generative coordinate: the surrogate evaluates the decoded molecule rather than regressing directly on $\boldsymbol{z}$.

We sample from Equation 14 using preconditioned Crank–Nicolson (pCN) proposals (Cotter et al., 2013). For chain c, let $\beta_c \in (0, 1]$ denote its proposal scale. Given the current state $\boldsymbol{z}$, we propose

$$\boldsymbol{z}' = \sqrt{1 - \beta_c^2}\, \boldsymbol{z} + \beta_c \boldsymbol{\xi}, \qquad \boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d). \tag{15}$$

Because pCN is reversible with respect to p_0, the Gaussian prior and proposal terms cancel from the Metropolis–Hastings ratio. The resulting acceptance probability is

$$A_t(\boldsymbol{z}, \boldsymbol{z}') = \min\left\{1, \frac{a_t(h_{\boldsymbol{\psi}}(\boldsymbol{z}'))}{a_t(h_{\boldsymbol{\psi}}(\boldsymbol{z}))}\right\}. \tag{16}$$

Parallel chains propose molecules enriched for high improvement probability, from which we retain only valid, previously unevaluated candidates.

Batch selection. To obtain the final evaluation batch, we score the generated candidates using the same PFN predictive distribution and retain those with the largest upper-confidence values,

$$\mathrm{UCB}_t(\boldsymbol{x}) = \mu_t(\boldsymbol{x}) + \kappa \sigma_t(\boldsymbol{x}), \tag{17}$$

where $\mu_t(\boldsymbol{x})$ and $\sigma_t(\boldsymbol{x})$ are the predictive mean and standard deviation under $q_{\boldsymbol{\omega}}(y \mid \rho(\boldsymbol{x}), \mathcal{C}_t^\rho)$. Thus, a single in-context predictive distribution plays two complementary roles: its improvement probabilities guide molecular generation, while its uncertainty-aware scores select the candidates sent to the oracle. The complete sampling procedure and implementation details are given in Appendix B.

3.3 Molecular continued pretraining

The pretrained PFN is exposed to a broad distribution of generic regression problems, whereas deployment in molecular LSBO involves structured molecular objectives and contexts concentrated around promising regions of the search space. We address this mismatch through continued pretraining on molecular, BO-oriented regression tasks, aiming to better align the predictor with its deployment distribution while retaining the broad inductive biases of the original checkpoint.

Synthetic molecular tasks. We construct molecular regression episodes from molecular scoring functions evaluated over a common support $\mathcal{M}$, given by the standardized ChEMBL collection used by GuacaMol (Brown et al., 2019). The library combines molecular properties, RDKit descriptors, and similarities to ZINC reference molecules. These functions define training targets and are distinct from the surrogate representation r. Importantly, the GuacaMol objectives used for downstream evaluation are excluded from the library. See Appendix C.7 for library construction and exclusion rules.

For each task τ, we sample K_τ base scores without replacement and combine them through a randomly sampled function C_τ,

$$F_\tau(\boldsymbol{x}) = C_\tau\left(s_{j_1}(\boldsymbol{x}), \ldots, s_{j_{K_\tau}}(\boldsymbol{x})\right). \tag{18}$$

The combiner family includes arithmetic, geometric, and power means, random MLPs, formula trees, and sampled GP functions. Combiners whose outputs are not naturally bounded are monotonically calibrated to $[0, 1]$. This task generation strategy is akin to that used in TabICLv2 (Qu et al., 2026).

To make the resulting datasets more representative of BO histories, we bias sampling toward high-value molecules. For $\boldsymbol{x}_m \in \mathcal{M}$, we define

$$p_{\tau,m} = \frac{\exp\{F_\tau(\boldsymbol{x}_m)/T_\tau\}}{\sum_{j=1}^{|\mathcal{M}|} \exp\{F_\tau(\boldsymbol{x}_j)/T_\tau\}}, \tag{19}$$

where T_τ controls the sampling bias, with lower values concentrating on high-scoring molecules. We then partition the sampled molecules into context and query sets, using query labels only for training. The same construction applies to molecular or latent surrogate representations. It captures the high-value bias of BO histories but does not reproduce a full rollout, see Appendix C for details.

Embedding-guided task selection. To favor synthetic objectives that the PFN processes similarly to molecular reference tasks, we follow Regenwetter et al. (2026) and represent each task through the hidden states it induces on a shared molecular panel. The panel and context–query partition are fixed across tasks, so only the task-specific labels change. Let $\boldsymbol{\omega}_0$ denote the frozen pretrained PFN weights, and let $\boldsymbol{h}^\tau_{\boldsymbol{\omega}_0,i} \in \mathbb{R}^{512}$ denote the hidden representation of query molecule i conditioned on the labeled context of task τ. We represent task τ by mean-pooling these query representations,

$$\boldsymbol{e}_\tau = \frac{1}{|\mathcal{Q}|} \sum_{i \in \mathcal{Q}} \boldsymbol{h}^\tau_{\boldsymbol{\omega}_0,i} \in \mathbb{R}^{512}. \tag{20}$$

We train an XGBoost classifier on these embeddings to distinguish molecular reference objectives, used only for task-bank selection, from synthetic tasks. Importantly, these reference objectives are disjoint from the downstream GuacaMol objectives used in experiments. We retain the N_{sel} synthetic tasks with the highest predicted reference-class probabilities and sample their CPT episodes using the procedure above. See Appendix C.5 for complete embedding construction and selection procedure.

Continued-pretraining objective. Starting from the official PFN weights $\boldsymbol{\omega}_0$, we retain TabPFN's bar-distribution regression parameterization, which represents continuous targets through probability masses over ordered output intervals, and add L2-SP regularization toward the pretrained parameters (Xuhong et al., 2018). For the model's L intervals, let $p_{\boldsymbol{\omega}}(\ell \mid \boldsymbol{u}, \mathcal{C})$ denote the predicted probability mass of interval ℓ for query $\boldsymbol{u}$ and context $\mathcal{C}$, and let $k(y) \in \{1, \ldots, L\}$ identify the interval containing target y. We minimize the corresponding interval cross-entropy,

$$\mathcal{L}_{\text{CPT}}(\boldsymbol{\omega}) = \mathbb{E}_{(\mathcal{C}_\tau, \mathcal{Q}_\tau)}\left[-\frac{1}{|\mathcal{Q}_\tau|} \sum_{(\boldsymbol{u},y) \in \mathcal{Q}_\tau} \log p_{\boldsymbol{\omega}}(k(y) \mid \boldsymbol{u}, \mathcal{C}_\tau)\right] + \frac{\lambda_{\text{SP}}}{2} \|\boldsymbol{\omega} - \boldsymbol{\omega}_0\|_2^2. \tag{21}$$

The expectation is over synthetic molecular episodes, with the predictive loss averaged over query rows within each episode and across episodes. The training configuration and optimization schedule are reported in Appendix C. The resulting PFN weights remain fixed during downstream BO.

3.4 Specialization to latent inputs

Direct latent prediction is recovered by setting $r = e_\psi$ in the common surrogate formulation (Equation 10). In this configuration, denoted PILBO-L, the PFN receives VAE coordinates directly and models the latent objective introduced in Section 2.1. For a candidate $\boldsymbol{z}$, prediction takes the form $q_{\boldsymbol{\omega}}(y \mid \boldsymbol{z}, \mathcal{C}_t^{e_\psi})$. This is the direct in-context analogue of conventional LSBO: the VAE provides both the search space and surrogate representation, and a pretrained PFN replaces the GP.

Latent-space optimization. We embed the PFN within the same local latent-space optimization pipeline used by conventional GP-based LSBO baselines. At each round, candidate latent coordinates are generated around promising observations, following TuRBO (Eriksson et al., 2019; Maus et al., 2022), and scored using predictive samples from $q_{\boldsymbol{\omega}}(y \mid \boldsymbol{z}, \mathcal{C}_t^r)$. The selected latent coordinates are decoded into molecules, which are evaluated and added to the history. The search procedure follows

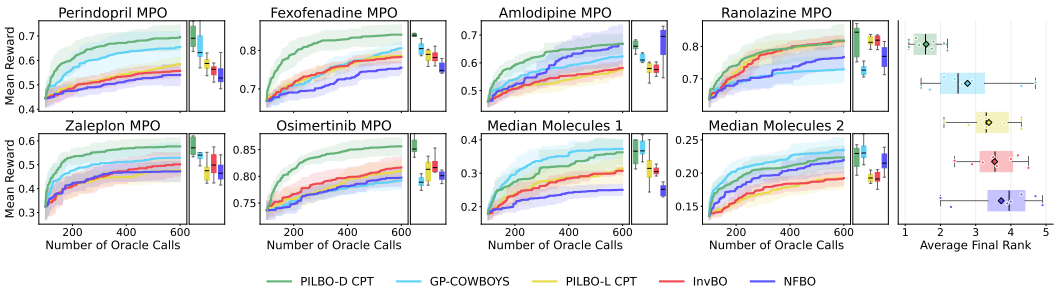

Figure 2: **Molecular optimization results.** Mean ± std best-observed reward and terminal-distribution summaries across ten seeds on 8 GuacaMol objectives.

InvBO (Chu et al., 2024) and is detailed in Appendix D.1, while the PFN context construction and latent-coordinate conventions are described in Appendix B.4.

However, this preserves the standard LSBO pipeline and therefore also its latent-space regression problem. The decoder mediates the map from z to objective value, while VAE updates and latent inversion can change the coordinates assigned to previously observed molecules and the induced latent objective. Continued pretraining (Section 3.3) is performed on a fixed latent representation, and therefore targets a geometry that may differ from the one presented to the PFN during optimization.

4 Related Work

Latent-space Bayesian optimization. LSBO uses continuous generative representations to make optimization over structured objects tractable (Gomez-Bombarelli et al., 2018). Much of the subsequent work addresses the relation between latent geometry and the objective. Weighted retraining emphasizes promising observations (Tripp et al., 2020), while LOLBO combines local search with generative-model adaptation (Maus et al., 2022). CoBO aligns latent distances with objective information (Lee et al., 2023); InvBO improves reconstruction through latent inversion (Chu et al., 2024); and NF-BO uses autoregressive normalizing flows to reduce reconstruction-induced mismatch (Lee et al., 2025). COWBOYS takes a different approach by separating molecular prediction from latent generation (Moss et al., 2025). Our study examines this representation choice for pretrained in-context surrogates. Of note, recent regret analysis of guided diffusion provides a complementary perspective on generator-based optimization over structured inputs (Adachi et al., 2026).

Prior-data fitted networks and adaptation. PFNs learn to approximate posterior prediction under a distribution of training tasks (Muller et al., 2021). Their development into tabular foundation models has enabled strong general-purpose classification and regression without task-specific weight updates (Hollmann et al., 2025; Grinsztajn et al., 2026; Qu et al., 2026). PFNs have also been applied to molecular-property prediction, where the choice of molecular representation is important: Ben Hicham et al. (2026) show that learned molecular embeddings and two-dimensional descriptors can outperform conventional fingerprints. Beyond the input representation, PFNs can be specialized before deployment through continued training, using real data (Garg et al., 2025) or target-like synthetic tasks (Regenwetter et al., 2026), in contrast to dataset-specific fine-tuning on deployment data (Rubachev et al., 2025). We similarly continue pretraining offline on synthetic molecular tasks designed to approximate LSBO and test whether this transfers to adaptively collected BO histories.

In-context learning for BO. The success of PFNs in prediction has spurred growing interest in in-context approaches to black-box optimization (Muller et al., 2023), including tabular foundation models coupled with specialized search procedures (Yu et al., 2026). Other methods amortize entropy-search acquisitions (Rakotoarison et al., 2026) or introduce sequential optimization priors (Li et al., 2025). A complementary direction learns policies that map observations directly to queries. Such policies may be trained from scratch (Maraval et al., 2023; Zhang et al., 2025; 2026; Blumer et al., 2026), or built on top of a pretrained tabular foundation model (Huang et al., 2026). We instead retain a predictive PFN and an explicit acquisition rule, and investigate how their effectiveness depends on molecular representation and domain-specific pretraining within LSBO.

5 Experiments

Baselines. We compare PILBO-D+CPT with the direct latent-space configuration PILBO-L+CPT. We also include GP-COWBOYS, using raw-pCN generation with qUCB (Moss et al., 2025), as

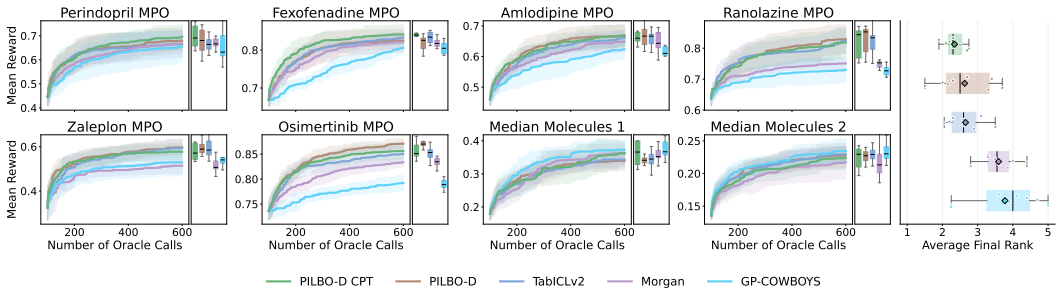

Figure 3: **Backbone and representation ablations.** Comparison of PILBO-D+CPT, PILBO-D, a TabICLv2-based variant, a Morgan-input variant, and GP-COWBOYS on eight GuacaMol objectives. Curves show mean best-observed reward over ten seeds with one-standard-deviation bands; adjacent boxplots summarize final scores at 600 oracle calls. The right panel reports average final ranks.

well as two state-of-the-art LSBO methods: NF-BO (Lee et al., 2025) and InvBO (Chu et al., 2024). GP-COWBOYS uses Morgan features with a Tanimoto kernel and joint batch acquisitions, whereas the decoupled PFN configuration uses molecular descriptors and pointwise UCB. Appendix D records the interfaces and method-specific search and VAE-update settings.

Implementation details. We evaluate on eight GuacaMol objectives (Brown et al., 2019). Crucially, none of these held-out objectives is included in the continued-pretraining task distribution (see Appendix C.7 for details). Unless otherwise stated, PILBO uses TabPFN-3 as its pretrained in-context surrogate. Each run starts with 100 evaluated molecules, acquires batches of five, and stops at 600 oracle calls, including initialization. The generator is a pretrained SELFIES VAE retrained after every 50 newly evaluated molecules (Maus et al., 2022). We report mean $\pm$ std of the best-observed reward over 10 seeds. See Appendix D for details on tasks, training, and evaluation.

5.1 Results

Pretrained PFNs provide strong surrogates for LSBO. Figure 2 displays the results. The decoupled formulation (PILBO-D+CPT) clearly outperforms GP-COWBOYS overall, while the direct latent formulation (PILBO-L+CPT) remains competitive, ranking ahead of InvBO and NF-BO in aggregate across the eight tasks. More broadly, the PFN-based configurations are competitive with or outperform established LSBO baselines across the benchmark. For completeness, Table S1 reports final performance for additional baselines, including LOLBO, CoBO, standard LSBO, and random search, which also rank below the PFN configurations. These findings are consistent with the strong small-data performance of tabular foundation models: unlike a GP fitted independently for each objective, a pretrained PFN can transfer inductive structure learned across many regression problems. One difference from GP-COWBOYS is the surrogate representation: our main PFN uses compact RDKit2D descriptors rather than 2,048-dimensional Morgan fingerprints, better matching TabPFN-3's default feature regime (Grinsztajn et al., 2026). We revisit this choice in Section 5.2.

Decoupling prediction from generation further improves in-context LSBO. PILBO-D+CPT substantially outperforms PILBO-L+CPT, echoing the benefit of separating prediction from generation observed for GP-based COWBOYS (Moss et al., 2025). Beyond decoupling itself, this gap may reflect how well the surrogate representation matches the inductive biases of the pretrained PFN: molecular descriptors expose chemically meaningful features, whereas direct prediction requires learning the objective over a reconstruction-trained generative latent space whose geometry is mediated by the decoder. This interpretation is consistent with recent molecular property-prediction results showing that tabular foundation model performance depends strongly on the representation, with RDKit2D descriptors and property-oriented learned embeddings substantially outperforming several alternative representations (Ben Hicham et al., 2026). Our findings extend this observation to BO, where surrogate inputs are generated adaptively along the optimization trajectory, with previous observations shaping subsequent candidates and their input distribution.

5.2 Ablations

We next disentangle the main design choices behind PILBO-D+CPT, focusing on the contribution of continued pretraining and the sensitivity to adaptation and representation choices (see Figure 3).

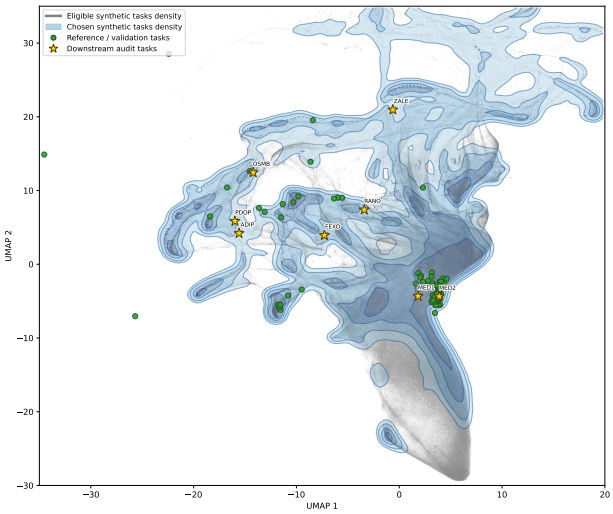

Figure 4: **Task resemblance in the PFN representation.** UMAP projection of the 512-dimensional task embeddings used for synthetic-objective selection. Each task is represented by mean-pooled PFN query hidden states on a shared molecular panel with a fixed context–query split. Grey and blue contours show eligible and selected synthetic objectives, respectively; green circles denote reference/validation tasks and yellow stars downstream GuacaMol objectives used only for evaluation. The selected objectives concentrate near molecular reference tasks in PFN representation space, confirming the intended effect of the selection procedure.

Effect of continued pretraining. The separation between the CPT and official checkpoints is larger early in optimization and narrows toward the final budget, with the same trend in mean seed rank (Table S2). This behavior is consistent with the Bayesian interpretation of PFNs: pretraining induces an implicit prior over regression tasks, and CPT modifies that prior toward molecular LSBO. Its effect should therefore be strongest with little task-specific evidence and diminish as context grows.

Sensitivity to CPT duration. Because longer continued pretraining need not improve downstream optimization, we select the CPT duration using two additional held-out GuacaMol objectives outside the main benchmark. We compare checkpoints after 5K, 10K, 15K, and 20K updates, retaining 15K, which gives the best aggregate validation performance, for all subsequent experiments (Appendix C.6, Figure S3). Differences across checkpoints remain modest and non-monotonic, with no systematic benefit from extending CPT further. This is consistent with previous reports that continued pretraining for tabular foundation models can trade off domain adaptation against preservation of useful pretrained structure (Garg et al., 2025; Kolberg et al., 2025).

Sensitivity to backbone and molecular representation. Figure 3 also varies the two main modeling choices underlying the decoupled surrogate, PILBO-D+CPT. Replacing TabPFN-3 with TabICLv2 tests dependence on the tabular foundation-model backbone, while replacing RDKit2D descriptors with Morgan fingerprints tests dependence on the molecular representation. TabICLv2 leads to only a moderate degradation, suggesting that the benefit of the decoupled formulation is not specific to TabPFN-3. Morgan fingerprints lead to a larger drop on several objectives, confirming that the representation matters, even though this version remains competitive and still outperforms GP-COWBOYS, which also uses Morgan fingerprints.

5.3 Offline-to-deployment mismatch in continued pretraining

Section 5.2 showed that the selected synthetic objectives lie near molecular reference tasks in the PFN embedding space (Figure 4), yet CPT provides only limited gains. This resemblance is measured on a fixed set of molecules shared across objectives (Section 3.3). Yet, during BO, the model predicts scores for newly generated molecules using observations collected along the search trajectory. We examine how these differences may limit transfer from synthetic pretraining to downstream optimization.

Different molecules and sampling procedures. CPT repeatedly samples molecules from a fixed ChEMBL pool, assigning scores under different synthetic objectives (Figure S1). BO instead generates candidates through the decoder. Across the BO trajectories for the held-out downstream tasks, only six of the 44,988 distinct post-initialization molecules appear in the CPT pool. Most molecules evaluated during BO were therefore absent from CPT, although they may still resemble training molecules in scaffold, fingerprint, or RDKit2D space. The way examples are selected also differs. CPT favors molecules with high synthetic scores, whereas BO uses previous observations to decide which molecules to evaluate next. Each new evaluation then enters the context used for subsequent predictions. The context and future queries are therefore linked by the search history, a dependence absent from the CPT sampling scheme. Appendix A.2 provides further diagnostics.

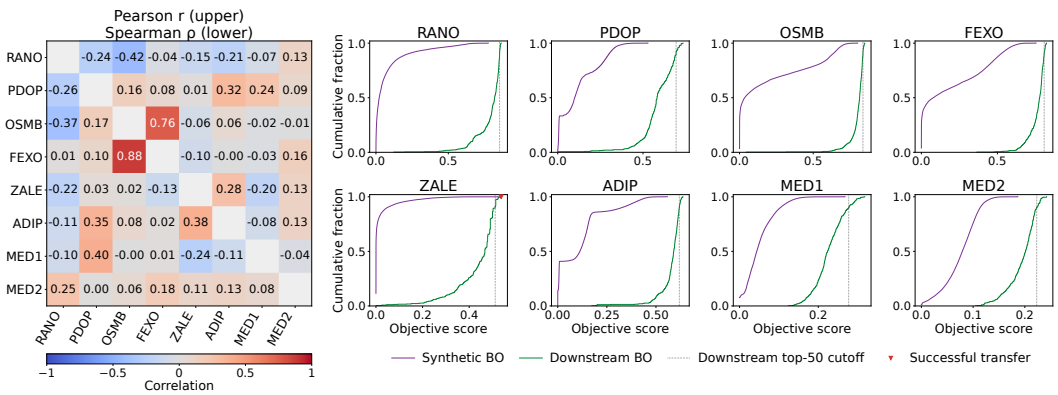

Figure 5: **Offline task similarity does not imply value transfer under BO.** Left: Pearson (upper triangle) and Spearman (lower triangle) correlations between downstream objectives on the fixed ChEMBL pool. Right: empirical cumulative score distributions for molecules found by optimizing embedding-selected synthetic objectives or the downstream objective. Both are scored using the same downstream objective in each panel. Dashed lines mark the score of the 50th-best downstream BO discovery. The red marker identifies the only synthetic-BO discovery reaching this cutoff.

Transfer between optimization objectives. Objectives can rank molecules similarly on the original pool without favoring the same discoveries during BO. For example, OSMB and FEXO have Spearman correlation 0.883 on the pretraining pool (Figure 5, left). To examine transfer from synthetic objectives, we run BO on tasks selected to resemble downstream objectives through embedding similarity or score correlation. We then evaluate the discovered molecules using the downstream objectives themselves. This downstream-informed selection and the resulting BO trajectories are used only for analysis, not to train the evaluated checkpoints. Although nearly all synthetic searches improve their own initial best score, their discoveries rarely achieve high downstream scores. For each downstream task, we use the score of the 50th-best molecule found by BO optimizing that task directly as a comparison threshold. Among the 413,402 distinct molecules from the embedding-selected searches, none reaches this threshold on seven of the eight tasks, and only one does so for Zaleplon (Figure 5, right). Thus, resemblance on the original pool does not ensure that optimizing a synthetic objective discovers molecules with high downstream scores.

Together, these results expose a deployment gap in both the molecular inputs $p(\boldsymbol{x})$ and the task-dependent relationships $p(y \mid \boldsymbol{x})$ encountered during optimization, helping explain why successful offline task matching can coexist with only limited CPT gains. Recent work further suggests that tabular in-context generalization depends strongly on retrieving and aggregating relevant context examples (Shaheen et al., 2026), providing a complementary explanation for why similarity on a fixed molecular support may not transfer to the adaptively collected contexts encountered during BO.

6 Conclusion

We introduced PILBO, an in-context LSBO framework combining pretrained PFN surrogates with latent-space molecular generation. Our results show that pretrained PFNs can transfer useful predictive structure to molecular LSBO, providing a strong alternative to fitting a surrogate anew for each objective. Decoupling prediction from latent generation is particularly effective, while the off-the-shelf PFN already accounts for much of the observed gain. CPT provides more limited additional improvements, even when constructed from synthetic tasks selected to resemble molecular reference objectives. Overall, our findings point to in-context learning as a promising foundation for more transferable and sample-efficient LSBO.

Limitations and future work. Our study restricts in-context LSBO to pretrained predictive surrogates combined with explicit acquisition rules, leaving end-to-end optimization policies outside its scope. Recent frameworks instead learn policies directly from optimization rollouts (Zhang et al., 2026), exposing the model during training to the adaptive, policy-induced data distributions that our CPT analysis suggests are difficult to reproduce offline. Extending this direction to LSBO is promising, although learning policies over its high-dimensional continuous search spaces remains challenging.

REPRODUCIBILITY STATEMENT

We provide the source code for the proposed method and scripts for reproducing our experiments in an anonymous repository at https://anonymous.4open.science/r/icl_lsbo-29C9. The appendix provides the hyperparameters used in our experiments and additional implementation details.

ETHICS STATEMENT

Our work develops methods for sample-efficient optimization over structured molecular spaces, with the potential to reduce the number of expensive evaluations required during molecular design. The experiments in this paper are limited to benchmark objectives and do not establish efficacy, safety, synthesizability, or clinical value of generated molecules. As with other molecular optimization methods, the same techniques could also be applied to undesirable objectives. Human oversight and appropriate scientific and safety constraints are therefore required when deploying such methods beyond benchmark settings.

AI USE STATEMENT

In this work, we used generative AI tools to assist with the implementation of methods, create and edit software code, assist with drafting and editing parts of the manuscript, and improve its readability and formatting. We have reviewed all AI-assisted work. AI-assisted code was reviewed and tested by the authors, and AI-assisted manuscript text was checked against the underlying results and relevant literature. We take responsibility for the final content of this work, including text, claims, code, and other artifacts produced with the aid of generative AI.

REFERENCES

Masaki Adachi, Anita Yang, Yakun Wang, and Song Liu. Regret analysis of guided diffusion for black-box optimization over structured inputs. *arXiv preprint arXiv:2605.10385*, 2026.

Karim K. Ben Hicham, Jan G. Rittig, Martin Grohe, and Alexander Mitsos. Tabular foundation models for in-context prediction of molecular properties. *arXiv preprint arXiv:2604.16123*, 2026.

Nicolas Samuel Blumer, Julien Martinelli, and Samuel Kaski. In-context black-box optimization with unreliable feedback. *arXiv preprint arXiv:2605.06187*, 2026.

Nathan Brown, Marco Fiscato, Marwin H. S. Segler, and Alain C. Vaucher. GuacaMol: Benchmarking models for de novo molecular design. *Journal of Chemical Information and Modeling*, 2019.

Jaewon Chu, Jinyoung Park, Seunghun Lee, and Hyunwoo J. Kim. Inversion-based latent bayesian optimization. In *Advances in Neural Information Processing Systems*, 2024.

Simon L. Cotter, Gareth O. Roberts, Andrew M. Stuart, and David White. MCMC methods for functions: Modifying old algorithms to make them faster. *Statistical Science*, 2013.

David Eriksson, Michael Pearce, Jacob Gardner, Ryan D. Turner, and Matthias Poloczek. Scalable global optimization via local bayesian optimization. In *Advances in Neural Information Processing Systems*, 2019.

Anurag Garg, Muhammad Ali, Noah Hollmann, Lennart Purucker, Samuel Müller, and Frank Hutter. Real-tabPFN: Improving tabular foundation models via continued pre-training with real-world data. In *1st ICML Workshop on Foundation Models for Structured Data*, 2025.

Roman Garnett. *Bayesian optimization*. Cambridge University Press, 2023.

Rafael Gomez-Bombarelli, Jennifer N. Wei, David Duvenaud, Jose Miguel Hernandez-Lobato, Benjamin Sanchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alan Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 2018.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin, Magnus Bühler, Anurag Garg, et al. TabPFN-3: Technical report. *arXiv preprint arXiv:2605.13986*, 2026.

Nate Gruver, Samuel Stanton, Nathan Frey, Tim G. J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion. In *Advances in Neural Information Processing Systems*, 2024.

Noah Hollmann, Samuel Muller, Lennart Purucker, Arjun Krishnakumar, Max Korfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 2025.

Daolang Huang, Zhuoyue Huang, Conor Hassan, Luigi Acerbi, Samuel Kaski, and Tom Rainforth. Efficient adaptive data acquisition via pretrained belief representations. *arXiv preprint arXiv:2606.25197*, 2026.

Christopher Kolberg, Jules Kreuer, Jonas Huurdeman, Sofiane Ouaari, Katharina Eggensperger, and Nico Pfeifer. TabPFN-Wide: Continued pre-training for extreme feature counts. *arXiv preprint arXiv:2510.06162*, 2025.

Seunghun Lee, Jaewon Chu, Sihyeon Kim, Juyeon Ko, and Hyunwoo J. Kim. Advancing bayesian optimization via learning correlated latent space. In *Advances in Neural Information Processing Systems*, 2023.

Seunghun Lee, Jinyoung Park, Jaewon Chu, Minseo Yoon, and Hyunwoo J. Kim. Latent bayesian optimization via autoregressive normalizing flows. In *International Conference on Learning Representations*, 2025.

Diantong Li, Kyunghyun Cho, and Chong Liu. None to optima in few shots: Bayesian optimization with mdp priors. *arXiv preprint arXiv:2511.01006*, 2025.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Alex Labach, Jesse Cresswell, Keyvan Golestan, Guangwei Yu, Anthony L Caterini, and Maks Volkovs. Tabdpt: Scaling tabular foundation models on real data. *Advances in Neural Information Processing Systems*, 2026.

Alexandre Maraval, Matthieu Zimmer, Antoine Grosnit, and Haitham Bou Ammar. End-to-end Meta-Bayesian Optimisation with Transformer Neural Processes. *Advances in Neural Information Processing Systems*, 2023.

Natalie Maus, Haydn Jones, Juston Moore, Matt J. Kusner, John Bradshaw, and Jacob Gardner. Local latent space bayesian optimization over structured inputs. In *Advances in Neural Information Processing Systems*, 2022.

Henry B. Moss, Sebastian W. Ober, and Tom Diethe. Return of the latent space cowboys: Re-thinking the use of vaes for bayesian optimisation of structured spaces. In *International Conference on Machine Learning*, 2025.

Samuel Muller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. *arXiv preprint arXiv:2112.10510*, 2021.

Samuel Muller, Matthias Feurer, Noah Hollmann, and Frank Hutter. PFNs4BO: In-context learning for Bayesian optimization. In *International Conference on Machine Learning*, 2023.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. *arXiv preprint arXiv:2602.11139*, 2026.

Herilalaina Rakotoarison, Steven Adriaensen, Tom Viering, Carl Hvarfner, Samuel Müller, Frank Hutter, and Eytan Bakshy. α-PFN: Fast entropy search via in-context learning. In *International Conference on Machine Learning*, 2026.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Lyle Regenwetter, Rosen Yu, Cyril Picard, and Faez Ahmed. Engineering regression without real-data training: Domain adaptation for tabular foundation models using multi-dataset embeddings. *arXiv preprint arXiv:2603.04692*, 2026.

Ivan Rubachev, Akim Kotelnikov, Nikolay Kartashev, and Artem Babenko. On finetuning tabular foundation models. *arXiv preprint arXiv:2506.08982*, 2025.

Nour Shaheen, Junwei Ma, Alex Labach, Frank Hutter, Valentin Thomas, and Anthony L. Caterini. Understanding the surprising generalization properties of tabular foundation models. *arXiv preprint arXiv:2608.17957*, 2026.

Marcelo DT Torres, Yimeng Zeng, Fangping Wan, Natalie Maus, Jacob Gardner, and Cesar de la Fuente-Nunez. A generative artificial intelligence approach for peptide antibiotic optimization. *Nature Machine Intelligence*, 2026.

Austin Tripp, Erik Daxberger, and Jose Miguel Hernandez-Lobato. Sample-efficient optimization in the latent space of deep generative models via weighted retraining. In *Advances in Neural Information Processing Systems*, 2020.

Li Xuhong, Yves Grandvalet, and Franck Davoine. Explicit inductive bias for transfer learning with convolutional networks. In *International Conference on Machine Learning*, 2018.

Rosen Ting-Ying Yu, Cyril Picard, and Faez Ahmed. GIT-BO: High-dimensional bayesian optimization with tabular foundation models. In *The Fourteenth International Conference on Learning Representations*, 2026.

Xinyu Zhang, Daolang Huang, Samuel Kaski, and Julien Martinelli. PABBO: Preferential Amortized Black-Box Optimization. *International Conference on Learning Representations*, 2025.

Xinyu Zhang, Conor Hassan, Julien Martinelli, Daolang Huang, and Samuel Kaski. In-context multi-objective optimization. In *International Conference on Learning Representations*, 2026.

Zilong Zhao, Robert Birke, and Lydia Y Chen. Tabula: Harnessing language models for tabular data synthesis. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2025.

APPENDIX OVERVIEW

- **Appendix A** reports additional experimental results and computational analyses, including the full baseline comparison, early-versus-final CPT performance, runtime measurements, and additional deployment diagnostics.
- **Appendix B** details the optimization algorithms and surrogate interfaces, including the pCN construction, the full decoupled optimization loop, predictive preprocessing, and the direct latent-input configuration.
- **Appendix C** describes synthetic-task construction and continued pretraining, including the molecular support and objective families, episode sampling, CPT configuration, embedding-guided task selection, checkpoint validation, and leakage controls.
- **Appendix D** provides the baseline configurations and definitions of the molecular benchmark objectives used in the experiments.

A ADDITIONAL RESULTS AND COMPUTATIONAL COST

Table S1: **Final performance on the eight GuacaMol tasks.** Best-observed reward at 600 oracle calls, reported as mean ± sample standard deviation over ten seeds. Methods are ordered by aggregate performance across tasks, and bold indicates the highest mean for each task.

Task	PILBO-D+CPT	GP-COWBOYS	PILBO-L+CPT	InvBO	LOLBO	NFBO	CoBO	LSBO	Random
RANO	**0.8181 ± 0.0483**	0.7297 ± 0.0398	0.8155 ± 0.0157	0.8167 ± 0.0166	0.7947 ± 0.0435	0.7676 ± 0.0374	0.7523 ± 0.0462	0.7719 ± 0.0377	0.6922 ± 0.0300
PDOP	**0.6963 ± 0.0476**	0.6546 ± 0.0643	0.5851 ± 0.0286	0.5570 ± 0.0247	0.5683 ± 0.0289	0.5409 ± 0.0437	0.5572 ± 0.0294	0.5437 ± 0.0469	0.4977 ± 0.0293
OSMB	**0.8562 ± 0.0158**	0.7921 ± 0.0151	0.8107 ± 0.0203	0.8166 ± 0.0211	0.8081 ± 0.0209	0.7973 ± 0.0159	0.7968 ± 0.0145	0.7845 ± 0.0197	0.7627 ± 0.0140
FEXO	**0.8411 ± 0.0254**	0.8058 ± 0.0165	0.7858 ± 0.0160	0.7833 ± 0.0153	0.7934 ± 0.0280	0.7551 ± 0.0143	0.7693 ± 0.0188	0.7459 ± 0.0153	0.7071 ± 0.0175
ZALE	**0.5769 ± 0.0361**	0.5291 ± 0.0614	0.4758 ± 0.0469	0.5015 ± 0.0489	0.4983 ± 0.0276	0.4723 ± 0.0419	0.4974 ± 0.0429	0.4543 ± 0.0359	0.4305 ± 0.0301
ADIP	**0.6689 ± 0.0338**	0.6231 ± 0.0312	0.5772 ± 0.0237	0.5818 ± 0.0251	0.5739 ± 0.0380	0.6679 ± 0.0686	0.5677 ± 0.0436	0.5387 ± 0.0239	0.5271 ± 0.0188
MED1	0.3624 ± 0.0365	**0.3727 ± 0.0269**	0.3144 ± 0.0440	0.3071 ± 0.0117	0.2937 ± 0.0163	0.2503 ± 0.0168	0.2991 ± 0.0306	0.2536 ± 0.0345	0.1801 ± 0.0180
MED2	0.2235 ± 0.0172	**0.2344 ± 0.0187**	0.1926 ± 0.0162	0.1926 ± 0.0093	0.1987 ± 0.0127	0.2197 ± 0.0200	0.1917 ± 0.0101	0.1646 ± 0.0124	0.1618 ± 0.0055
Overall mean ↑	**0.630403**	0.592696	0.569645	0.569560	0.566141	0.558890	0.553945	0.532158	0.494884
Sum ↑	**5.043226**	4.741567	4.557161	4.556484	4.529128	4.471121	4.431561	4.257262	3.959072
Rank by sum ↓	**1**	2	3	4	5	6	7	8	9

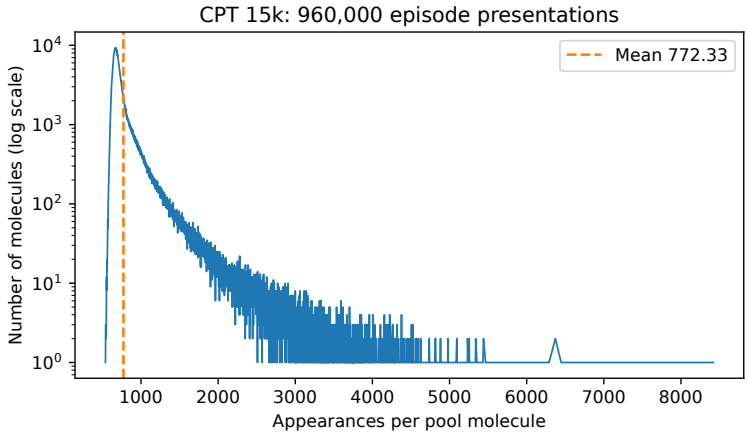

Figure S1: **Molecular reuse across 15K CPT updates.** Measured appearance-count distribution over the finite pool, using the complete 960,000-presentation schedule. The mean is 772.33 appearances per molecule.

Table S2: **CPT is more beneficial early in optimization.** Aggregate comparison of the continued-pretrained and official TabPFN-3 checkpoints at 150 and 600 oracle calls. Mean rank computed using the same number of baselines as in Figure 2.

	150 calls		600 calls	
Configuration	Mean	Mean rank	Mean	Mean rank
Official checkpoint	0.5365	2.01	0.6292	2.14
CPT	0.5402	1.79	0.6304	2.04

A.1 Runtime analysis

Table S3: **End-to-end optimization time.** Runtime in hours, reported as mean $\pm$ one sample standard deviation across runs.

Method	Hours
PILBO-D+CPT	1.9256 ± 0.2992
GP-COWBOYS	1.5204 ± 0.5343
PILBO-L+CPT	0.0818 ± 0.0189
InvBO	0.0336 ± 0.0127
LOLBO	0.0184 ± 0.0029
CoBO	0.0189 ± 0.0064
NFBO	0.5026 ± 0.0651
LSBO	0.0150 ± 0.0030
Random	0.0125 ± 0.0005

The decoupled methods are slower than approaches that operate directly in latent space because candidate scoring requires returning to molecular space (Table S3). Each pCN proposal must first be decoded into a molecule and represented by molecular features before its acquisition value can be evaluated. This overhead is shared by PILBO-D+CPT and GP-COWBOYS, whereas direct latent-space methods can score candidate coordinates without this repeated decode-and-featurize step.

Table S4 decomposes this cost for the two decoupled methods. Molecular decoding alone accounts for 78.05 minutes (67.6%) of PILBO-D+CPT runtime and 88.98 minutes (97.5%) for GP-COWBOYS. Across the 100 BO rounds, the methods decode approximately 101,266 and 132,362 proposals per run, respectively. The remaining cost includes molecular feature construction, surrogate evaluation, and MCMC updates.

PILBO-D+CPT incurs a larger residual cost than GP-COWBOYS, with 37.49 versus 2.24 minutes outside decoding. The search repeatedly evaluates small candidate batches against contexts of at most 600 observations. In this regime, cached GP prediction is relatively inexpensive, whereas the PFN still requires transformer inference in addition to molecular feature computation.

Table S4: **Runtime breakdown for the decoupled methods.** Mean minutes per run across eight tasks and ten seeds per task. Percentages refer to mean total runtime. The "other" category includes molecular feature construction, surrogate evaluation, and MCMC updates.

Component	PILBO-D+CPT	GP-COWBOYS
Molecular decoding	78.05 (67.6%)	88.98 (97.5%)
Other work	37.49 (32.4%)	2.24 (2.5%)
Total	115.54	91.22

A.2 Additional deployment diagnostics

Table S5 complements the molecular-support analysis in Section 5.3 with additional statistics on molecular overlap within CPT and between CPT and downstream BO trajectories.

Table S5: **Identity-level support diagnostics.** Overlap statistics for CPT episodes and audited downstream BO trajectories. Exact molecular identity is the unit of comparison; these quantities do not measure scaffold or descriptor similarity.

Quantity	Reported value
Mean exact overlap between pairs of 1,024-row CPT episodes	0.914 molecules
Audited BO molecular identities absent from the CPT pool	99.987%

We further test whether embedding-guided selection improves matching along an actual BO trajectory (Section 3.3 and Appendix C.5). At several optimization rounds, we compare the observed, context, and query score distributions with their closest synthetic reference from either the embedding-selected or a random collection (Figure S2). Embedding-guided selection does not consistently produce a closer rollout-conditioned match.

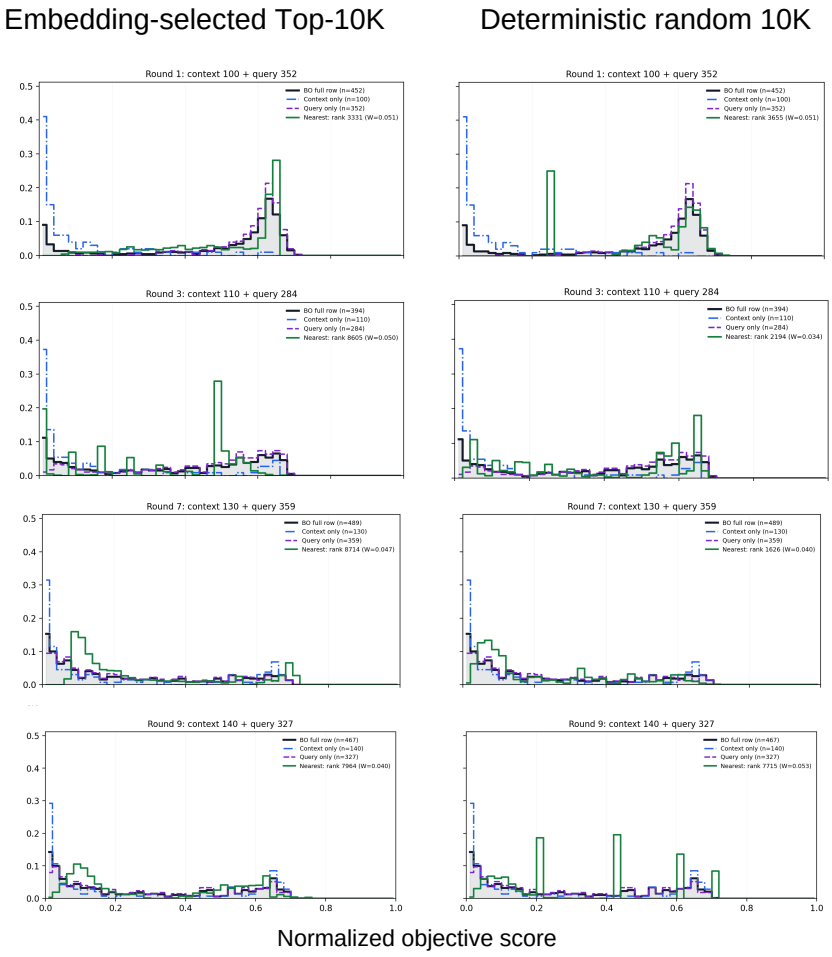

Figure S2: **Fixed-panel task matching does not consistently improve rollout-conditioned matching.** At several rounds of an example downstream BO trajectory, we compare the normalized objective-score distributions of accumulated BO observations (black), context rows (blue), and query rows (purple) with the closest synthetic reference (green). Left: embedding-selected synthetic objectives. Right: a deterministic random collection of synthetic objectives. Fixed-panel selection does not consistently yield a closer reference along the adaptive trajectory.

B ALGORITHMS AND SURROGATE INTERFACES

B.1 CONDITIONAL GENERATION AND PCN ACCEPTANCE

Our conditional target follows COWBOYS (Moss et al., 2025),

$$\pi_t(\mathrm{d}\boldsymbol{z}) \propto a_t(h_{\boldsymbol{\psi}}(\boldsymbol{z}))\, p_0(\mathrm{d}\boldsymbol{z}), \tag{S1}$$

where a_t is the probability of improvement defined in Equation 13.

Fix a BO round and proposal scale β, and let $G_\beta(\boldsymbol{z}, \mathrm{d}\boldsymbol{z}')$ denote the pCN proposal kernel. For

$$\boldsymbol{z}' = \sqrt{1-\beta^2}\boldsymbol{z} + \beta\boldsymbol{\xi}, \qquad \boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d), \tag{S2}$$

the kernel is reversible with respect to the Gaussian latent prior p_0 (Cotter et al., 2013),

$$p_0(\mathrm{d}\boldsymbol{z})G_\beta(\boldsymbol{z}, \mathrm{d}\boldsymbol{z}') = p_0(\mathrm{d}\boldsymbol{z}')G_\beta(\boldsymbol{z}', \mathrm{d}\boldsymbol{z}). \tag{S3}$$

Writing

$$w_t(\boldsymbol{z}) = a_t(h_{\boldsymbol{\psi}}(\boldsymbol{z}))\,, \qquad \pi_t(\mathrm{d}\boldsymbol{z}) = Z_t^{-1} w_t(\boldsymbol{z}) p_0(\mathrm{d}\boldsymbol{z}), \tag{S4}$$

the Metropolis–Hastings ratio therefore simplifies to

$$\frac{\pi_t(\mathrm{d}\boldsymbol{z}')G_\beta(\boldsymbol{z}', \mathrm{d}\boldsymbol{z})}{\pi_t(\mathrm{d}\boldsymbol{z})G_\beta(\boldsymbol{z}, \mathrm{d}\boldsymbol{z}')} = \frac{w_t(\boldsymbol{z}')}{w_t(\boldsymbol{z})}. \tag{S5}$$

Hence, the Gaussian prior and proposal terms cancel, giving the acceptance probability used in our method,

$$A_t(\boldsymbol{z}, \boldsymbol{z}') = \min\left\{1, \frac{a_t(h_{\boldsymbol{\psi}}(\boldsymbol{z}'))}{a_t(h_{\boldsymbol{\psi}}(\boldsymbol{z}))}\right\}. \tag{S6}$$

The incumbent threshold is kept fixed during each candidate-generation sweep, and invalid decoded molecules are assigned zero weight. Proposal-scale adaptation, finite chains, candidate deduplication, and the subsequent UCB selection make the overall procedure an approximate finite-sample realization of this conditional target; the complete algorithm is given below.

B.2 FULL DESCRIPTOR-SPACE OPTIMIZATION LOOP

Algorithm 1 Decoupled in-context LSBO with pCN candidate generation

Require: VAE decoder $h_{\boldsymbol{\psi}}$, Gaussian prior p_0, frozen PFN $q_{\boldsymbol{\omega}}$, descriptor map ρ
Require: 100 initial evaluated molecules with aligned records; total budget 600; batch $B = 5$

Initialize ten chain scales $\beta_c \leftarrow 0.1$
while $|\mathcal{D}| < 600$ **do**
  Form the full descriptor context $\mathcal{C} \leftarrow \{(\rho(\boldsymbol{x}_i), y_i)\}$; set $y^\star \leftarrow \max_i y_i$
  Start every chain at the stored aligned raw latent of the incumbent
  Set candidate pool $\mathcal{P} \leftarrow \varnothing$ and sweep counter $s \leftarrow 0$
  repeat
    $s \leftarrow s + 1$
    for each chain c **do**
      Propose $\boldsymbol{z}'_c$ (Equation 15); decode $\boldsymbol{x}'_c \leftarrow h_{\boldsymbol{\psi}}(\boldsymbol{z}'_c)$
      Compute its probability of improvement and accept or reject (Equation 16)
      if the proposal is accepted **then**
        $\boldsymbol{z}_c \leftarrow \boldsymbol{z}'_c$
      end if
      Adapt β_c toward acceptance 0.234, retaining $0 < \beta_c \leq 1$
      if the proposal is accepted, valid, and canonically novel **then**
        Add $(\boldsymbol{z}'_c, \boldsymbol{x}'_c, \rho(\boldsymbol{x}'_c))$ to $\mathcal{P}$
      end if
    end for
  until $s \geq 100$ and $|\mathcal{P}| \geq B$, or $s = 1000$
  Require $|\mathcal{P}| \geq B$; choose B molecules by pointwise UCB with $\kappa = 1$
  Evaluate those molecules; append values and aligned records to $\mathcal{D}$
end while
return Best evaluated molecule

Chain scales are carried across BO rounds, while chain states are reinitialized at the incumbent. Candidate molecules are deduplicated against both the observation history and the current pool. For each generated molecule, we retain the latent coordinate that produced it rather than re-encoding the decoded molecule, since the two coordinates need not coincide under a deterministic decoder.

Candidate generation and batch selection are performed separately. Improvement probabilities guide the pCN chains, after which the final batch is selected by pointwise UCB using the PFN marginal predictive means and variances. This selection does not model joint posterior dependence between batch members, beyond excluding exact duplicate molecules.

B.3 Predictive representation and preprocessing

The main surrogate uses 212 of the 217 ordered RDKit2D descriptors, excluding five that overlap with benchmark objective components (MolLogP, TPSA, NumAromaticRings, RingCount, and BertzCT). Its inference configuration is E1: one view with squashing, ordinal handling, and quarter-SVD preprocessing, an identity target transform, no feature shift, and seed 1. The official and CPT comparisons use identical preprocessing. The descriptor definitions remain fixed, while data-dependent preprocessing is applied consistently within each BO round.

B.4 Direct latent-input configuration

The natural latent PFN uses 256-dimensional SELFIES-VAE codes. A local trust-region host generates candidate coordinates around promising observations, retrieves recent and high-value context points, and performs its predictive-sampling acquisition. VAE updates and InvBO-style inversion may then change the coordinates presented to the PFN.

For a given molecule $\boldsymbol{x}$, encoder and inversion inputs are different maps:

$$\boldsymbol{z}_t^{\mathrm{enc}}(\boldsymbol{x}) = e_{\boldsymbol{\psi}_t}(\boldsymbol{x}), \qquad \boldsymbol{z}_t^{\mathrm{inv}}(\boldsymbol{x}) \in \arg\min_{\boldsymbol{z}} \{-\log p_{\boldsymbol{\psi}_t}(\boldsymbol{x} \mid \boldsymbol{z})\}. \tag{S7}$$

Inversion can improve reconstruction (Chu et al., 2024), while also altering the input distribution seen by a predictor pretrained on $\boldsymbol{z}_0^{\mathrm{enc}}$. This distinction is especially important when the generative model itself changes. The observation value remains $f(\boldsymbol{x})$, not necessarily $f(h_{\boldsymbol{\psi}_t}(\boldsymbol{z}_t^{\mathrm{enc}}(\boldsymbol{x})))$.

C Synthetic molecular tasks and continued pretraining

C.1 Molecular support and base-score catalogue

The shared support is the standardized ChEMBL training collection used by GuacaMol, with $M = 1{,}272{,}822$ molecules (Brown et al., 2019). Table S6 summarizes the 100-score catalogue used for synthetic label generation.

Table S6: Base-score catalogue for molecular synthetic tasks. These 100 functions generate synthetic labels (not to be confused with the 212 RDKit input descriptor columns).

Family	Count	Content
Core scores	8	Albuterol and mestranol similarity; celecoxib, thiothixene, and troglitazone rediscovery; two molecular-formula isomer scores; QED.
RDKit descriptors	27	Calibrated physicochemical, topological, count, complexity, drug-likeness, and structural-alert scores.
ZINC similarities	47	Fingerprint similarity to individual reference molecules.
ZINC median similarities	18	Median similarity tasks based on pairs of reference molecules.
Total	100	

C.2 Objective families

For each task, choose $K \in \{6, \dots, 10\}$ scores without replacement and write $\boldsymbol{s}(\boldsymbol{x}) = [s_{j_1}(\boldsymbol{x}), \dots, s_{j_K}(\boldsymbol{x})]^\top$. The reported bounded aggregators have the forms

$$C_{\mathrm{lin}}(\boldsymbol{s}) = \sum_{j=1}^{K} w_j s_j, \qquad C_{\mathrm{geo}}(\boldsymbol{s}) = \prod_{j=1}^{K} s_j^{w_j}, \quad w_j \geq 0, \quad \sum_j w_j = 1,$$
$$C_{\mathrm{pow}}(\boldsymbol{s}; p) = \left(\frac{1}{K} \sum_{j=1}^{K} s_j^p \right)^{1/p}, \qquad p \neq 0. \tag{S8}$$

Random smooth MLPs, formula trees, and GP function draws provide nonlinear alternatives. Let $\widetilde{C}_\tau$ denote a raw unbounded combiner and c_τ its monotone output calibration; then $F_\tau = c_\tau \circ \widetilde{C}_\tau \circ \boldsymbol{s}$. These combiners broaden the target family without supplying new molecules.

C.3 Without-replacement sampling and context/query construction

The temperature $T_\tau > 0$ in Equation 19 controls how strongly a synthetic episode favors high-scoring molecules. Low temperature favors the elite part of the pool; high temperature approaches uniform sampling. For a Gumbel-top-n implementation, independent $G_m \sim \mathrm{Gumbel}(0, 1)$ produce the index set

$$I_\tau = \mathrm{TopK}_{1024} \left\{ \frac{F_\tau(\boldsymbol{x}_m)}{T_\tau} + G_m : 1 \leq m \leq M \right\}. \tag{S9}$$

This represents weighted sampling without replacement; the selected rows are not 1,024 independent draws from the Boltzmann probabilities. After selection, an independent joint permutation is applied to feature rows and targets. Context size is then sampled uniformly from $\{100, \dots, 600\}$, and the remaining rows become queries.

Algorithm 2 Synthetic episode construction for a chosen representation

Require: Molecular pool $\mathcal{M}$, base-score catalogue, representation r, generator configuration
1: Draw $K \in \{6, \dots, 10\}$ and K distinct base scores
2: Draw combiner parameters and form calibrated objective F_τ
3: Evaluate F_τ on $\mathcal{M}$; draw temperature $T_\tau \sim \mathrm{LogUniform}(0.01, 10)$
4: Select 1,024 distinct indices using weighted without-replacement sampling
5: Jointly permute pairs $(r(\boldsymbol{x}_i), F_\tau(\boldsymbol{x}_i))$ for the selected indices
6: Draw $n_{\mathrm{ctx}} \sim \mathrm{Uniform}\{100, \dots, 600\}$
7: Use the first n_{ctx} rows as context; use the remaining features as queries
8: Compute predictive loss against the withheld query targets

For the descriptor method, $r = \rho$. A latent version instead uses a fixed encoder representation during CPT.

C.4 SELECTED DESCRIPTOR-SPACE CONFIGURATION

Table S7: Configuration associated with the selected descriptor-space CPT checkpoint.

Component	Setting
Starting model	Official TabPFN-3 regression weights
Surrogate input	212 RDKit2D descriptors, excluding five that overlap with downstream objective components
Episode geometry	1,024 distinct molecules; context uniformly 100–600 rows; remaining rows queried
Predictive objective	$\mathcal{L}_{\mathrm{CE}}$
Checkpoint anchor	L2-SP coefficient 0.003, fixed
Optimizer	Muon with cautious weight decay 0.01
Schedule	100-step warm-up to 10^{-5}, then cosine decay to 10^{-7}
Stabilization	Global gradient clipping at 5; BF16 arithmetic
Training scale	15,000 updates; global batch 64 tasks; 960,000 episodes
BO weights	Frozen after offline checkpoint selection

C.5 EMBEDDING-GUIDED SYNTHETIC TASK SELECTOR

The selector adapts the dataset-embedding approach of Regenwetter et al. (2026) to the molecular task generator. All tasks use the same panel of 10,000 molecules, with fixed, disjoint index sets $\mathcal{I}_{\mathrm{ctx}}$ and $\mathcal{Q}$ containing 600 context and 9,400 query rows, respectively. For task τ, let $y_{\tau,j}$ denote its objective value at molecule $\boldsymbol{x}_j$. The labeled selector context is

$$\mathcal{C}_\tau^{\mathrm{sel}} = \{(\rho(\boldsymbol{x}_j), y_{\tau,j}) : j \in \mathcal{I}_{\mathrm{ctx}}\}. \tag{S10}$$

The shared query-feature matrix is

$$\mathbf{U}_{\mathcal{Q}} = [\rho(\boldsymbol{x}_i)^\top]_{i\in\mathcal{Q}} \in \mathbb{R}^{9400\times d_\rho}. \tag{S11}$$

The molecular features and row ordering are shared across tasks. Only the context labels change.

Let $\mathbf{H}_{\boldsymbol{\omega}_0}^{(\ell_{\mathrm{emb}})}$ denote the query-representation map of the frozen PFN at extraction layer ℓ_{emb}. The representation associated with query i is

$$\mathbf{h}_{\tau,i} = \left[\mathbf{H}_{\boldsymbol{\omega}_0}^{(\ell_{\mathrm{emb}})}\left(\mathbf{U}_{\mathcal{Q}} \mid \mathcal{C}_\tau^{\mathrm{sel}}\right)\right]_i \in \mathbb{R}^{512}, \tag{S12}$$

where $[\cdot]_i$ extracts the hidden vector corresponding to query i. We average these vectors to obtain the task embedding,

$$\boldsymbol{e}_\tau = \frac{1}{|\mathcal{Q}|}\sum_{i\in\mathcal{Q}}\left[\mathbf{H}_{\boldsymbol{\omega}_0}^{(\ell_{\mathrm{emb}})}\left(\mathbf{U}_{\mathcal{Q}} \mid \mathcal{C}_\tau^{\mathrm{sel}}\right)\right]_i \in \mathbb{R}^{512}. \tag{S13}$$

Thus, the embedding summarizes how the PFN represents the shared query molecules after conditioning on the task-specific context. Query objective values $\{y_{\tau,i} : i \in \mathcal{Q}\}$ are not supplied.

We train an XGBoost classifier on these task embeddings. Let $\mathcal{T}_{\mathrm{ref}}$ contain the 71 positive reference tasks: 25 real/base tasks, 33 single-reference ZINC similarity tasks, and 13 ZINC median-reference tasks. Let $\mathcal{T}_{\mathrm{neg}}$ contain the 250 deterministic synthetic negative tasks. The classifier training set is

$$\mathcal{D}_{\mathrm{sel}} = \{(\boldsymbol{e}_\tau, 1) : \tau \in \mathcal{T}_{\mathrm{ref}}\} \cup \{(\boldsymbol{e}_\tau, 0) : \tau \in \mathcal{T}_{\mathrm{neg}}\}. \tag{S14}$$

These binary labels identify reference versus synthetic tasks and are distinct from the molecular objective values used in the PFN context. The reference-task collection is also distinct from the 100-function catalogue used to generate synthetic objectives.

Writing $f_{\mathrm{XGB}}(\cdot;\widehat{\vartheta})$ for the fitted classifier's reference-class probability, we score each eligible synthetic task as

$$s_\tau = f_{\mathrm{XGB}}(\boldsymbol{e}_\tau;\widehat{\vartheta}) = \widehat{\mathbb{P}}_{\widehat{\vartheta}}(c = 1 \mid \boldsymbol{e}_\tau), \tag{S15}$$

where c is the task-class label. We retain the N_{sel} highest-scoring tasks from the eligible candidate bank $\mathcal{B}$, equivalently

$$\mathcal{B}_{\mathrm{sel}} \in \operatorname*{arg\,max}_{\substack{\mathcal{S}\subseteq\mathcal{B}\\ |\mathcal{S}|=N_{\mathrm{sel}}}} \sum_{\tau\in\mathcal{S}} s_\tau. \tag{S16}$$

The synthetic negatives are excluded from $\mathcal{B}$, and downstream evaluation objectives are excluded from the selector training set. The PFN remains frozen throughout embedding extraction, and the classifier is frozen after fitting. Algorithm 3 summarizes the selection procedure.

Algorithm 3 Embedding-guided curation of a synthetic task bank

Require: Frozen PFN with weights $\boldsymbol{\omega}_0$, extraction layer ℓ_{emb}
Require: Fixed 10,000-molecule panel with context and query index sets $\mathcal{I}_{\text{ctx}}$ and $\mathcal{Q}$ of sizes 600 and 9,400
Require: Reference tasks $\mathcal{T}_{\text{ref}}$ synthetic negatives $\mathcal{T}_{\text{neg}}$, and eligible bank $\mathcal{B}$, selection size N_{sel}
Construct the shared query-feature matrix $\mathbf{U}_{\mathcal{Q}}$ (Equation S11)
for each task $\tau \in \mathcal{T}_{\text{ref}} \cup \mathcal{T}_{\text{neg}}$ **do**
    Evaluate context labels and form $\mathcal{C}_{\tau}^{\text{sel}}$ (Equation S10)
    Extract query representations without query labels (Equation S12)
    Mean-pool the representations into $\boldsymbol{e}_{\tau}$ (Equation S13)
end for
Construct the labeled embedding dataset $\mathcal{D}_{\text{sel}}$ (Equation S14)
Fit XGBoost on $\mathcal{D}_{\text{sel}}$ to obtain $\widehat{\vartheta}$; freeze the fitted classifier
for each candidate task $\tau \in \mathcal{B}$ **do**
    Evaluate context labels and form $\mathcal{C}_{\tau}^{\text{sel}}$ (Equation S10)
    Compute $\boldsymbol{e}_{\tau}$ without query labels (Equation S13)
    Assign $s_{\tau} = f_{\text{XGB}}(\boldsymbol{e}_{\tau}; \widehat{\vartheta})$ (Equation S15)
end for
Select the N_{sel} highest-scoring tasks to form $\mathcal{B}_{\text{sel}}$ (Equation S16)
return $\mathcal{B}_{\text{sel}}$

The selector uses a fixed molecular panel and a 600-row context to make task embeddings comparable. CPT subsequently draws new episodes from the selected objectives, using Boltzmann-biased sampling over the full molecular pool and context sizes ranging from 100 to 600 (Appendix C). Thus, task selection does not restrict CPT molecules to the embedding panel.

C.6 Validation of the CPT checkpoint choice

To assess our choice of the 15K CPT checkpoint, we evaluate the same four checkpoints on two additional GuacaMol benchmarks: Valsartan SMARTS (VALT) and decoration hopping (DHOP) (Brown et al., 2019). Table S8 summarizes their component families, and Figure S3 compares the performance of the four checkpoints.

Table S8: Additional benchmarks used to assess the CPT checkpoint choice and their component families. PHCO denotes pharmacophore fingerprints; SMARTS specifies substructure patterns.

Task	Scoring function(s)	Modifier
Valsartan SMARTS (VALT)	$\text{SMARTS}(s_1, \text{true})$	None
	logP	$\text{Gaussian}(2.0165, 0.2)$
	TPSA	$\text{Gaussian}(77.04, 5)$
	Bertz complexity	$\text{Gaussian}(896.38, 30)$
Decoration hopping (DHOP)	$\text{SMARTS}(s_2, \text{true})$	None
	$\text{SMARTS}(s_3, \text{false})$	None
	$\text{SMARTS}(s_4, \text{false})$	None
	$\text{sim}(r_{\text{DHOP}}, \text{PHCO})$	$\text{Thresholded}(0.85)$

For the respective transformed component scores $u_j(\boldsymbol{x}), v_j(\boldsymbol{x}) \in [0, 1]$, the objectives are

$$f_{\text{VALT}}(\boldsymbol{x}) = \left\{ \prod_{j=1}^{4} u_j(\boldsymbol{x}) \right\}^{1/4}, \qquad f_{\text{DHOP}}(\boldsymbol{x}) = \frac{1}{4} \sum_{j=1}^{4} v_j(\boldsymbol{x}). \tag{S17}$$

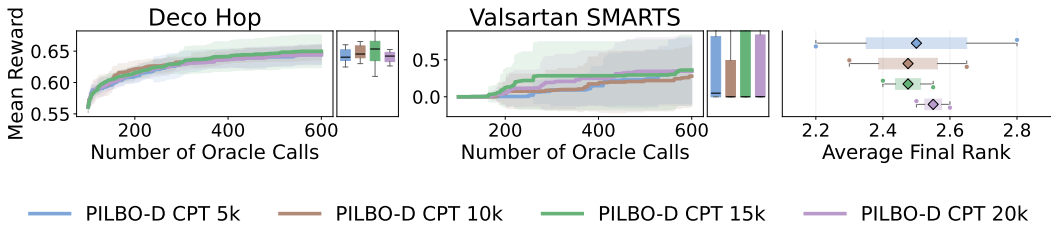

Figure S3: **CPT checkpoint validation.** Comparison of the 5K, 10K, 15K, and 20K CPT checkpoints on two additional GuacaMol objectives. Curves show mean best-observed reward over ten seeds with one-standard-deviation bands; adjacent boxplots summarize final scores at 600 oracle calls. The right panel reports average final ranks within this four-checkpoint comparison.

C.7 Exclusion rules and benchmark-feature access

We avoid data leakage in our pipeline at the level of synthetic objectives, surrogate inputs, and task selection.

- **Synthetic objectives.** Complete downstream objective specifications are never used to generate synthetic training labels, and individual molecular properties that also appear explicitly as components of the downstream benchmark objectives are excluded from the 100-score catalogue.
- **Surrogate inputs.** As mentioned in Appendix B.3, the RDKit2D molecular representation excludes MolLogP, TPSA, NumAromaticRings, RingCount, and BertzCT, leading to a 212-dimensional representation. These quantities were excluded because they appear explicitly in the downstream benchmark objectives (Appendix D.3). Removing them prevents the surrogate from directly observing target components, although correlated information may remain in the other descriptors.
- **Task selection.** The reference objectives and synthetic negatives used by the embedding-guided selector are disjoint from the downstream GuacaMol objectives. Query labels on the shared embedding panel are not provided to the PFN or classifier, and downstream labels are used only for post hoc evaluation and visualization, not for task selection, checkpoint selection, or preprocessing.

D Benchmark and evaluation details

D.1 Baselines and comparison configurations

Each run starts with 100 evaluated molecules and acquires batches of five until reaching 600 oracle calls, including initialization. We report results over 10 seeds. For all VAE-based methods, generative models are periodically retrained during optimization, after every 50 newly evaluated molecules, corresponding to every ten BO rounds.

GP-COWBOYS (Moss et al., 2025). GP-COWBOYS decouples molecular prediction from latent-space generation. A GP predicts objective values from molecular features, while a pretrained VAE supplies candidate molecules. The predictive probability of improvement defines a conditional generation target, which is sampled through pCN-based MCMC in the VAE latent space. In our comparison, the surrogate uses Morgan features with a Tanimoto kernel, candidate generation operates on raw latent coordinates, and the evaluation batch is selected using qUCB. The GP is updated as observations accumulate. Unlike the original COWBOYS implementation, we also periodically retrain the VAE, matching the update protocol used by the other VAE-based methods. We adopt this modification both for a fairer comparison and because it improves GP-COWBOYS performance in our experiments.

InvBO (Chu et al., 2024). Inversion-based latent Bayesian optimization augments LSBO with decoder inversion and potential-aware trust-region anchor selection. Inversion searches for a latent code that reconstructs an observed molecule, reducing the mismatch between the molecule whose objective was evaluated and the molecule represented by the surrogate input. This step uses the

decoder reconstruction objective and requires no additional oracle evaluations. Anchor selection combines the observed value of a candidate anchor with the potential of its surrounding trust region, estimated from posterior samples. A GP guides local candidate selection through Thompson sampling.

NF-BO (Lee et al., 2025). NF-BO replaces the VAE with SeqFlow, an autoregressive normalizing-flow model for molecular sequences. Its encoding map is injective and its decoder is a left inverse on encoded sequences, addressing the reconstruction mismatch between an observed molecule and its latent representation. A GP surrogate models objective values in this latent space. Candidate generation uses token-level adaptive perturbations, allocating exploration according to token importance, and Thompson sampling selects the next evaluations. Thus, NF-BO changes the generative representation as well as the latent candidate-generation procedure.

D.2 VAE RETRAINING SCHEDULE

The VAE is periodically updated using newly collected observations. For all configurations with VAE adaptation, retraining is scheduled after every 50 new oracle evaluations. Each update trains the encoder and decoder for two epochs using Adam with learning rate 10^{-3}, on the five most recent and 50 highest-scoring observations.

D.3 MOLECULAR BENCHMARK OBJECTIVES

Table S9 gives the component families for the eight current tasks: six MPO benchmarks and two median molecules benchmarks. Each task combines its component scores using a geometric mean after applying any specified modifiers (Brown et al., 2019). Figure S4 illustrates the component modifiers.

Table S9: Current MPO and median molecules suite and its component functions and modifiers. $\mathrm{sim}(r, \mathrm{FP})$ denotes Tanimoto similarity to reference molecule r using fingerprint FP. AP denotes atom-pair fingerprints; ECFP and FCFP denote the fingerprint families specified by the benchmark.

Task	Scoring function(s)	Modifier
Ranolazine	$\mathrm{sim}(\text{ranolazine}, \mathrm{AP})$ logP TPSA Number of fluorine atoms	$\mathrm{Thresholded}(0.7)$ $\mathrm{MaxGaussian}(7, 1)$ $\mathrm{MaxGaussian}(95, 20)$ $\mathrm{Gaussian}(1, 1)$
Perindopril	$\mathrm{sim}(\text{perindopril}, \mathrm{ECFP4})$ Number of aromatic rings	None $\mathrm{Gaussian}(2, 0.5)$
Osimertinib	$\mathrm{sim}(\text{osimertinib}, \mathrm{FCFP4})$ $\mathrm{sim}(\text{osimertinib}, \mathrm{ECFP6})$ TPSA logP	$\mathrm{Thresholded}(0.8)$ $\mathrm{MinGaussian}(0.85, 0.1)$ $\mathrm{MaxGaussian}(100, 10)$ $\mathrm{MinGaussian}(1, 1)$
Fexofenadine	$\mathrm{sim}(\text{fexofenadine}, \mathrm{AP})$ TPSA logP	$\mathrm{Thresholded}(0.8)$ $\mathrm{MaxGaussian}(90, 10)$ $\mathrm{MinGaussian}(4, 1)$
Zaleplon	$\mathrm{sim}(\text{zaleplon}, \mathrm{ECFP4})$ $\mathrm{isomer}(\mathrm{C_{19}H_{17}N_3O_2})$	None None
Amlodipine	$\mathrm{sim}(\text{amlodipine}, \mathrm{ECFP4})$ Number of rings	None $\mathrm{Gaussian}(3, 0.5)$
Median molecules 1	$\mathrm{sim}(\text{camphor}, \mathrm{ECFP4})$ $\mathrm{sim}(\text{menthol}, \mathrm{ECFP4})$	None None
Median molecules 2	$\mathrm{sim}(\text{tadalafil}, \mathrm{ECFP6})$ $\mathrm{sim}(\text{sildenafil}, \mathrm{ECFP6})$	None None

For component scores $u_j(\boldsymbol{x}) \geq 0$, a geometric-mean task has the generic form

$$f(\boldsymbol{x}) = \left\{ \prod_{j=1}^{J} u_j(\boldsymbol{x}) \right\}^{1/J} . \tag{S18}$$

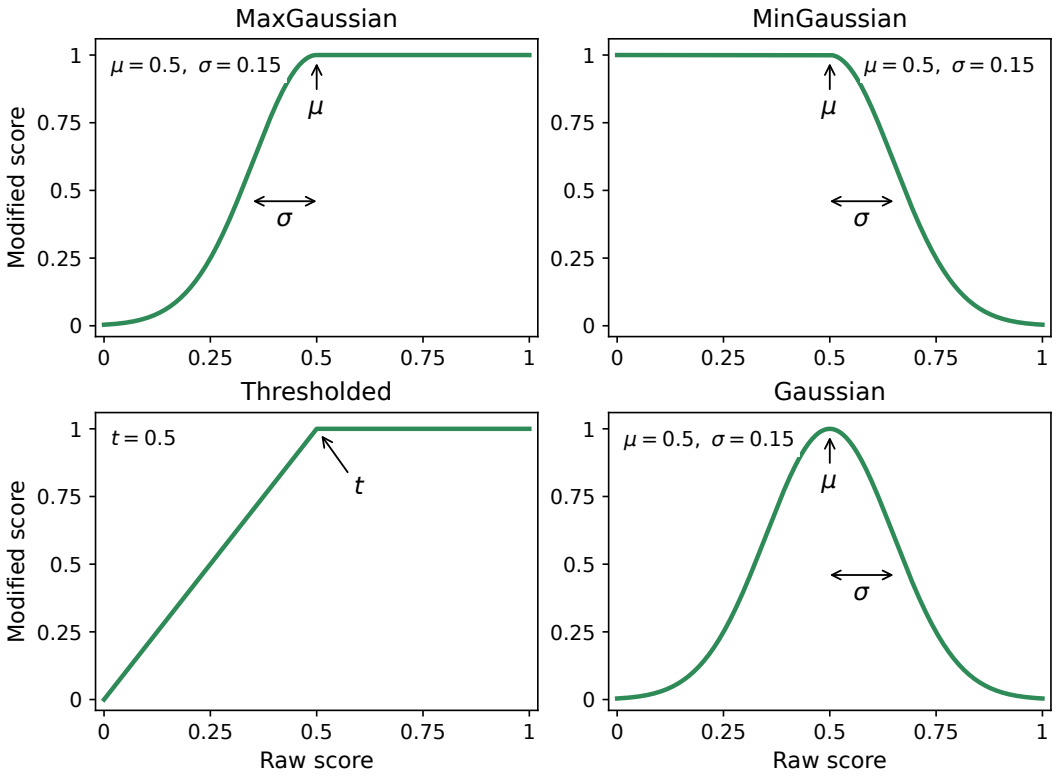

Figure S4: **Score modifiers used in the benchmark objectives** (Brown et al., 2019). Gaussian rewards proximity to a target μ. MinGaussian and MaxGaussian assign unit score to values $x \leq \mu$ and $x \geq \mu$, respectively, with Gaussian decay of width σ on the opposite side. Thresholded assigns $\min(x/t, 1)$ for nonnegative inputs. Curves use the illustrative parameters indicated in each panel; benchmark-specific parameters are listed in Tables S8 and S9.