MOLECULAR ACTIVE LEARNING: CAN LLMS HELP?

Anonymous authors
Paper under double-blind review

ABSTRACT

Drug discovery, and molecular discovery more broadly, can be framed as a sequential active learning problem —facing a candidate pool, strategies are designed to sequentially acquire molecules to assay, aiming to find the best molecule within the fewest rounds of trial and error. To automate this process, Bayesian optimization (BO) methods can mimic the approach of human medicinal chemists by constructing *representations* from existing knowledge, quantifying *uncertainty* for the predictions, and designing *acquisition* experiments that balance exploitation and exploration. Traditionally, these three stages are implemented using building blocks such as graph neural networks (GNN) as representations, variational inference (VI) or Gaussian process (GP) for uncertainty quantification, and analytical expressions as acquisition functions. To facilitate the integration of both domain-specific and general knowledge into various stages of this process, in this paper, we investigate which parts of this workflow can be augmented or replaced by large language models (LLM). To this end, we present **COLT** [1], a software library for **C**hemical **O**ptimization with **L**anguage- and **T**opology-based modules, and thoroughly benchmark the combination thereof. We found that *none* of the LLMs, no matter incorporated at what stage, can outperform the simple and fast Bayesian baseline with GNN and GP. As a remedy, we offer a new tuning recipe with direct preference optimization (DPO), where the optimization of synthetic properties can be used to increase the efficiency of the acquisition in real-world tasks.

The short answer to the question asked in the title is: *Not easily.*

1 INTRODUCTION: DRUG DISCOVERY AS SEQUENTIAL ACTIVE LEARNING

A drug discovery campaign—the endeavor to search, from the vast chemical universe, for a new chemical entity with some desired therapeutic efficacy—takes decades and billions of dollars and has its decision-making process traditionally reliant on human experts [1; 2]. When a human expert makes a decision to prioritize a certain compound(s) in the pre-clinical stage of a drug discovery project, their thought process, not without some oversimplification (regarding the multitask, constrained, and batched nature of the optimization), could be broken down as follows: first, an understanding of the current chemical space is constructed from existing medicinal chemistry data, as well as general knowledge distilled from years of training and experience (sometimes referred to as the *chemical intuition* [3]); second, this understanding is applied to the chemical space yet to be assayed, providing predictions with associated uncertainty; finally, by leveraging expectation and uncertainty, while balancing exploration vs. interpolation, she selects candidates to be synthesized and characterized, generating assay data to refine her belief. This process is carried out iteratively until a therapeutic candidate suitable for clinical trials is identified.

Bayesian optimization methods [4; 5] can work analogously to human experts when optimizing over the chemical space, with the aforementioned three stages corresponding to *representation*, *uncertainty quantification*, and *acquisition* in the active learning process. We can use graph neural networks (GNNs) [6; 7; 8; 9; 10; 11; 12] to represent the molecule (§ 2.1), variational inference (VI) [13] or Gaussian process (GP) regression [14; 15] to quantify the uncertainty of the predictions (§ 2.2), and analytical expressions as the acquisition functions (§ 2.3).

[1] code at: `https://anonymous.4open.science/r/colt-6B75/`

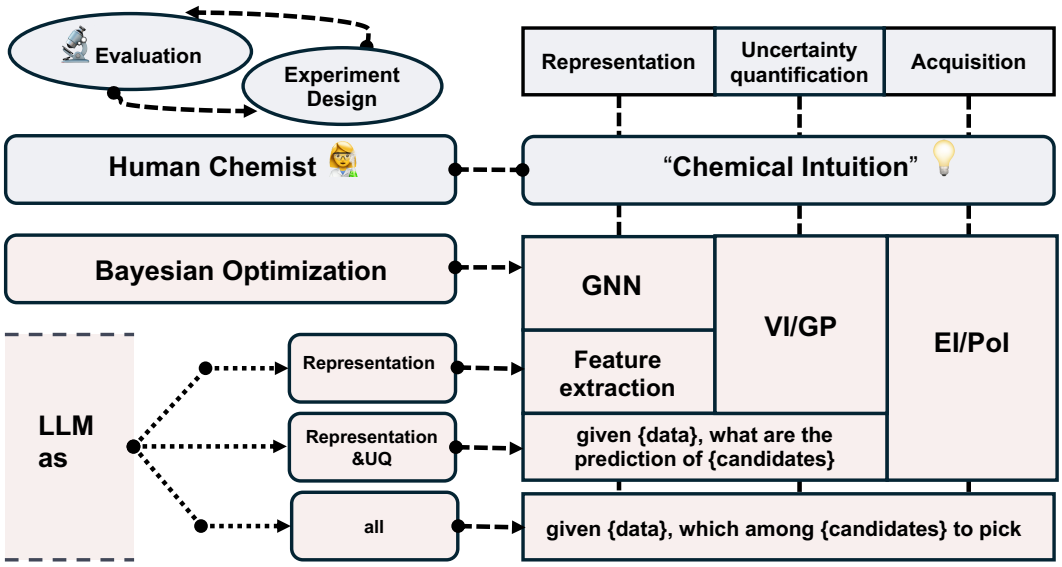


Figure 1: **Semantic illustration.** The experiment design in drug discovery can be broken down into the representation, uncertainty quantification, and acquisition stages, working analogously to the chemical intuition of human medicinal chemists. Bayesian optimization (§ 2) typically employs GNN as representation (§ 2.1), VI or GP (§ 2.2) as uncertainty quantification, and analytical functions (§ 2.3) for acquisition, which can be modularly replaced by LLMs using the example prompts.

Notably, the majority of early-stage drug discovery problems dwell in the *small-data* regime. Unlike the abundant cat pictures or blog posts readily scraped from the internet, in drug discovery, each data point is the result of costly and time-consuming wet lab experiments. Consequently, even the largest pharmaceutical companies rarely work with a candidate pool larger than 1 million compounds [16]—the number being smaller in magnitude for biotech startups. ChEMBL [17], the largest public chemical data repository, contains merely 15 million assays of various kinds over 1.8 million compounds. On one hand, this makes principled Bayesian models particularly well suited, thanks to their inherent regularization and uncertainty quantification. On the other hand, while working with graph-structured data provides useful inductive biases, incorporating these biases into such Bayesian, graph-based models remains challenging [18; 19], unlike LLMs, which can easily *absorb* language-structured knowledge by training on texts. In the chemistry domain, for instance, the success of LLMs has been demonstrated through numerous models fine-tuned for diverse modeling tasks [20; 21; 22; 23].

Main contributions. In this paper, we offer a comparison between the efficacy of *knowledge* versus *inductive bias* by evaluating the optimization efficiency of LLMs against that of BO with graph-based representation. First, we review the abstractions underlying the building blocks of Bayesian active learning and propose solutions to replace each of them with LLMs. To this end, we introduce a package for **C**hemical **O**ptimization with **L**anguage- and **T**opology-based models (COLT), which offers an easy-to-use API for molecular active learning. Using this program, we combinatorially benchmark the effect of LLMs when placed at various stages of active learning, and find that none of the solutions are as performant as the GNN + GP baseline. Offering a beacon of hope amidst the negative results, we provide a general tuning recipe based on *direct preference optimization* (DPO) [24], where learning occurs directly on the per-step *acquisition trajectory*, and find that tuning on synthetic datasets has a positive impact on optimizing real-world tasks.

Limitations. In this paper, we only consider the optimization on a finite candidate space, whereas the combinatorial chemical space is infinite and can be treated continuously [25]. Additionally, to enable rapid benchmarking with a limited computational footprint, we restrict our analysis to small datasets, resulting in high variance across the benchmark results. Similarly, we restrict our experiments to smaller, open-source LLMs.

2 BAYESIAN OPTIMIZATION (BO) FOR MOLECULAR ACTIVE LEARNING

First, we formalize the drug discovery campaign introduced in § 1. Given a finite pool of candidate chemical compounds, represented by chemical graphs $\mathcal{F} = \{\mathcal{G}_i, i = 1, 2, \dots N\}$, each associated with a *potency* $y(\mathcal{G}_i) \in \mathbb{R}$, of which a noisy observation is expensive to acquire, we are interested in finding the compound with the highest potency $\mathcal{G}^* = \text{argmax}_i\, y(\mathcal{G}_i)$ with fewest function evaulations. This can be achieved via *active learning*, starting with an empty portfolio $\mathcal{P} = \emptyset$, and in each round, choosing a compound from the candidate pool, $\mathcal{F}$, based on a decision informed by the current portfolio, $\hat{\mathcal{G}} = D(\mathcal{F}|\mathcal{P}), \hat{\mathcal{G}} \in \mathcal{F}$. Subsequently, we subtract this compound from the candidate pool to add it to the portfolio, $\mathcal{F} \leftarrow \mathcal{F} \setminus \{\hat{\mathcal{G}}\}, \mathcal{P} \leftarrow \mathcal{P} \cup \{\hat{\mathcal{G}}\}$.

Bayesian optimization (BO) is a powerful approach in active learning where the decision $D(\mathcal{F}|\mathcal{P})$ is constructed in a modular way. Firstly, we extract fixed D-dimensional features from the compounds to form a *representation*: $H = h(\mathcal{G}) \in \mathbb{R}^D$; secondly, this representation is used to construct a *predictive posterior distribution* with *uncertainty quantification*: $p(y|\mathcal{G}; \Theta), \Theta = \theta(H)$, whose parameters Θ is is mapped from the graph representations. Lastly, an *acquisition* function, α, is applied on this predictive posterior to form a score, on the basis of which a decision is made: $\hat{G} = D(\mathcal{F}|\mathcal{P}) = \text{argmax}_{\mathcal{G}\in\mathcal{F}}\, \alpha(p(y|\mathcal{G}))$.

In sum, the acquisition trajectory represents a sequential decision process, where the probability of a given trajectory can be written as:

$$p(\mathcal{P} = \{\mathcal{G}_i\}) = p(\mathcal{G}_0) \prod_{t=0}^{t=|\mathcal{F}|} p(\mathcal{G}_t|\{\mathcal{G}_{t=0,\cdots,t}\}) \tag{1}$$

$$= p(\mathcal{G}_0) \prod_{t=0}^{t=|\mathcal{F}|} \int \mathrm{d}\,\Theta p(\Theta|\{\mathcal{G}_{t=0,\cdots,t}\}))P[\alpha(p(y|\mathcal{G}_t, \Theta)) \geq \alpha(p(y|\mathcal{G}_i, \Theta)), \forall \mathcal{G}_i \in \mathcal{F}]., \tag{2}$$

where the first equality (1) stands generally for active learning strategies surveyed in this paper (§ 2 and § 3), and the second equality stands only for traditional Bayesian optimization (§ 2).

2.1 REPRESENTATION: GRAPH NEURAL NETWORKS (GNN)

Graph neural networks (GNNs) have emerged to be the modern workhorse for graph representation. A GNN can be most generally defined as one adopting a layer-wise updating scheme that aggregates representations from a node's neighborhood $\mathcal{N}(v)$ (based on the edges $\hat{A}_{uv}$) and updates its embedding:

$$\mathbf{X}'_v = \phi(\mathbf{X}_v, \rho(\mathbf{X}_u, \hat{A}_{uv}, u \in \mathcal{N}(v))), \tag{3}$$

where ϕ, ρ are the *update* and *aggregate* function, repsectively. Omitting the nonlinear transformation step ϕ, common to all neural network models, and assuming a convolutional *aggregate* function, $\rho = \text{SUM}$ or $\rho = \text{MEAN}$, a GNN layer is characterized by the aggregation/convolution operation that pools representations from neighboring nodes. This forms an intermediary representation $\mathbf{X}'$, which on a global level, with activation function σ and weights W, can be written as: $\mathbf{X}' = \sigma(\hat{A}\mathbf{X}W)$. The primary difference among architectures lies in the choice of effective adjacency matrix, $\hat{A}$. The most classical examples include: graph convolutional networks [6] (GCN), which normalize $\hat{A}$ by the node in-degree, $D_{ii} = \sum_j A_{ij}$, and graph attention networks [11] (GAT), which take $\hat{A}$ to be the attention score;

$$\hat{A}_{\texttt{GCN}} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}; \hat{A}_{\texttt{GAT},ij} = \text{Softmax}(\sigma(\text{NN}(\mathbf{X}_i||\mathbf{X}_j))); \tag{4}$$

Stacking GNN layers, we can get a representation of the graph

$$H = \mathbf{X}^{(l)} = \underbrace{\sigma(\hat{A}\sigma(\hat{A}\sigma(\hat{A}...\sigma(\hat{A}}_{l\ \texttt{times}}\mathbf{X}\underbrace{W...W)W)W)}_{l\ \texttt{times}}. \tag{5}$$

2.2 UNCERTAINTY QUANTIFICATION: VARIATIONAL INFERENCE (VI) AND GAUSSIAN PROCESSES (GP) REGRESSION

Bayesian neural networks. Under the Bayesian formalism, given sets of (graph, measurement) pairs as training data $\mathcal{D} = \{\mathcal{G}^{(i)}, y^{(i)}, i = 1, 2, 3, ..., n\}$, the probability distribution of the unknown

quantity of the measurement $y^{(n+1)}$ could be modelled with respect to the posterior distribution of the model $\mathcal{M}$ with parameters θ as:

$$p(y^*|\mathcal{M}, \mathcal{D}, \mathcal{G}^*) = \int p(y^*|\mathcal{G}^*, \Theta) P(\Theta|\mathcal{D}) \, \mathrm{d}\, \theta. \tag{6}$$

This integral, of course, is not tractable, and fully Bayesian methods using a Markov chain Monte Carlo-based sampling scheme can turn out to be prohibitively expensive with realistically sized datasets and models. We review two common techniques to approximate Equation 6.

Variational inference on the parameter space. Variational inference [13] turns the sampling problem into an optimization problem by assuming the distribution belongs to a specific family and and then optimizing its parameters to best approximate the true distribution. Concretely, on a parameter space [26], we assume such class of distribution to be a multivariate Gaussian with diagonal covariance matrix $q(\Theta) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$, whose parameters can be optimized by minimizing the Kullback-Leibler (KL) divergence between the variational and true Bayesian posterior (also known as variational free energy):

$$\boldsymbol{\mu}^*, \boldsymbol{\sigma}^* = \arg\min_{\boldsymbol{\mu}, \boldsymbol{\sigma}} \mathcal{D}_{\mathrm{KL}}[q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})||P(\boldsymbol{\theta}|\mathcal{D})] = \arg\min_{\boldsymbol{\mu}, \boldsymbol{\sigma}} \mathcal{D}_{\mathrm{KL}}[q(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\sigma})||P(\boldsymbol{\theta})] - \mathbb{E}_{\boldsymbol{\theta} \sim q(\boldsymbol{\mu}, \boldsymbol{\sigma})}[\log P(\mathcal{D}|\boldsymbol{\theta})], \tag{7}$$

Gaussian process (GP) regression with deep kernel learning (DKL) Within the deep kernel learning framework [27], a graph kernel can be defined by applying a standard kernel (such as the radial basis function, RBF) with parameters γ, to the output of GNNs.

$$k(\mathcal{G}, \mathcal{G}') = k_{\mathrm{RBF}}(\mathrm{GNN}(\mathcal{G}|\Theta), \mathrm{GNN}(\mathcal{G}'|\Theta), \gamma) \tag{8}$$

Equation 6 can then be written as:

$$p(\mathbf{y}^*|\mathcal{G}^*, \mathcal{D} = \{\mathcal{G}\}) \sim \mathcal{N}(\mathbb{E}[y^*], \mathrm{cov}(y^*)), \tag{9}$$

where:

$$\mathbb{E}[\mathbf{y}^*] = K(\mathcal{G}^*, \{\mathcal{G}\})[K(\{\mathcal{G}\}, \{\mathcal{G}\}) + \sigma_n^2 I]^{-1}\mathbf{y}; \tag{10}$$

$$\mathrm{cov}(\mathbf{y}^*) = K(\{\mathcal{G}^*\}, \{\mathcal{G}^*\}) - K(\{\mathcal{G}^*\}, \{\mathcal{G}\})[K(\{\mathcal{G}\}, \{\mathcal{G}\}) + \sigma_n^2 I]^{-1} K(\{\mathcal{G}\}, \{\mathcal{G}^*\}). \tag{11}$$

The neural network and the kernel parameters $\{\Theta, \gamma\}$ can be jointly optimized to produce a maximum likelihood fit to the dataset.

2.3 Acquisition functions

Once the model is trained, we can use the *predictive posterior*, $p(y^*|\mathcal{G}^*, \boldsymbol{\theta})$, of a given compound to prioritize compounds from a candidate pool by defining an *acquisition function* α and greedily selecting the candidate with the largest α for subsequent assaying. Popular choices include [28]:

Probability of Improvement (PoI),

$$\alpha_{\mathrm{PI}}(\mathcal{G}^*|\mathcal{D}, \Theta) = 1 - \Phi_{P(y^*|\mathcal{G}^*, \boldsymbol{\theta})}(\max(y)), \tag{12}$$

characterizes the probability of the best current value, where Φ denotes the CDF of the corresponding distribution, and $\max(y)$ is obtained within the training set $\mathcal{D} = \{\mathcal{G}_i, y_i\}$

Expected improvement (EI),

$$\alpha_{\mathrm{EI}}(\mathcal{G}^*|\mathcal{D}, \Theta) = \mathbb{E}_{P(y^*|\mathcal{G}^*, \boldsymbol{\theta})} \min\{\{y^*\} - \max(y), 0\}, \tag{13}$$

measures the expectation of improvement over the current best.

For tractable distributions such as the Gaussian family, these expressions can be evaluated analytically, though in general they can also be estimated via samples drawn from these distributions.

3 MODULARLY REPLACING BO COMPONENTS WITH LLMS

Large language models (LLMs)—neural networks composed primarily of transformer [29]—have shown surprising promise [30; 31; 32; 33] in representing and generating text-structured data and gained popularity very quickly in recent years. Their impressive generative ability led the field to describe them with personifying terms, such as *understanding*, and to employ them in seemingly impossible tasks, from reasoning [34] to planning [35].

In recent years, LLMs [31; 30] have been routinely incorporated into active learning pipelines, from hyperparameter search [36] to material design [37; 38]. Little attention, however, has been paid to dissecting which *stage* of the active learning process can LLMs be most effective in. To answer this question, and to compare the efficacy of LLMs with time-tested workflows, we propose three solutions to incorporate LLMs into molecular active learning by modularly replacing components outlined in § 2 by LLMs.

3.1 LLM AS REPRESENTATION

```
h = transformers.pipeline("feature-extraction")(molecule.smiles)
```

Replacing the GNNs (§ 2.1), one can use an LLM as a feature extractor to come up with the *representation* of molecules. The input of the LLM is a string representation of the graph, such as the SMILES string [39] common in molecular representation. This is consistent with the method Kristiadi et al. [38] used for material discovery.

Even before the era of LLMs, to represent molecules as strings and feeding them into transformer- or recurrent neural networks (RNN)-based models have long been used in various pipelines of molecular modeling, from property prediction to active learning [25], where pretraining on large ensembles of data has been proven as an effective avenue towards better performance [40]. We are interested in testing whether the added complexity and *knowledge* in LLM would help refine this representation.

3.2 LLMS AS REPRESENTATION AND UNCERTAINTY QUANTIFICATION

```
data = [molecule.smiles, str(molecule.y) for molecule in data]
prompt = f"""
Given a list of molecules with associated properties: {data},
what is the property of the molecule {new_molecule}?
"""
posterior_samples = [vllm.LLM().generate(prompt) for _ in range(N)]
```

The generation process of LLMs is intrinsically stochastic, and one can harvest the stochasticity of that process as a proxy of the uncertainty. Specifically, the predictions of a property of a molecule can be predicted by LLMs in a few-shot, in-context [41] manner, where a few examples are provided in the prompts, based on which a prediction of the desired property is made. This replaces both the *representation* (§ 2.1) and the *uncertainty quantification* (§ 2.2) stages of the Bayesian optimization. Ramos et al. [37] employed this method for quantifying the uncertainty of property predictions of materials. A natural challenge present here, as in all in-context learning methods, is the curse of the token limit, which allows very few samples to be included. When moving from the small-data to big-data regime, this can be solved via selecting in-context learning examples via similarity measures.

3.3 LLMS AS REPRESENTATION, UNCERTAINTY QUANTIFICATION, AND ACQUISITION

```
data = [molecule.smiles, str(molecule.y) for molecule in data]
candidates = [candidate.smiles for candidate in candidates]
prompt = f"""
In an active learning setting,
given a list of molecules with associated properties: {data},
which among the {candidates} to assay next to maximize the property
within the fewest rounds of assay?
"""
```

Finally, we can replace *all* building blocks introduced in § 2 with LLMs and let them make final choices of compounds to assay in the next round. When the pool of candidates is large, the length of the prompt will grow quickly. To enable token-efficient search among large candidate pools, we introduce a *tournament* search model, where we separate the candidates into groups, and iteratively compare the best candidate within groups. As such, if the token limit can originally fit only N candidates, the tournament search model can allow up to N^M candidates to be compared within NM prompts.

4 COLT: A LIBRARY FOR MOLECULAR ACTIVE LEARNING.

```
from colt import *
for representation in [GCN, LLMRepresentation]:
    for uncertainty in [VI, GP, LLMUncertainty]:
        for acquisition in [EI, LLMAcquisition]:
            model = acquisition(uncertainty(representation()))
            trial = Trial(model=model, data=ESOL(), steps=100)
```

To rapidly benchmark the three strategies illustrated in Section 3, and to provide a platform for medicinal chemists to seamlessly integrate both traditional and language-based active learning approaches into their workflows – thereby accelerating the design of life-saving therapeutics – we introduce COLT, a software package for chemical optimization using language- and topology-based methods. The above is a illustration of the COLT library interface used to generate the results in § 5, where *representation* (§ 2.1 or § 3.1), *uncertainty quantification* (§ 2.2 or § 3.2), and *acquisition* function (§ 2.3 or § 3.3) are modularly abstracted.

Speed. Designed for practical simulation, the efficiency in terms of wall time has been a focus of the design from the beginning. As such, in this PyTorch [42]-based package, the GNNs are implemented with generalized sparse matrix-matrix multiplication in deep graph library (DGL) [43], variational inference parallelized on GPU using Pyro [44] for variational inference, Gaussian processes regression with kernel interpolation [45] implemented in GPytorch [14], and fast deployment of LLMs with Huggingface [46] and vLLM [47].

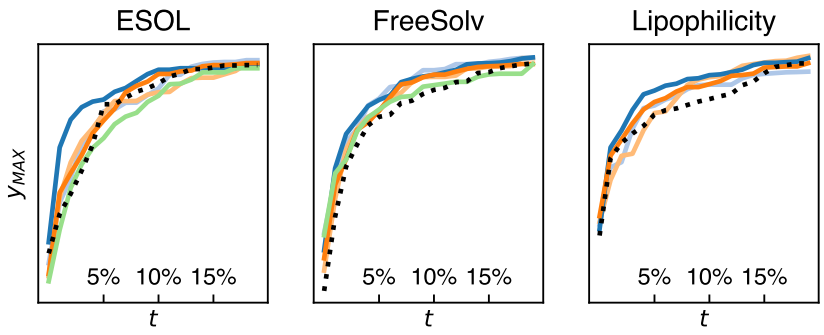
Similar packages. `GAUCHE` [48] performs Gaussian process regression using graph- and string-based kernels. `Lapeft-Bayesopt` [38] uses the representation from LLMs to perform Bayesian optimization to discover materials. COLT differs from these efforts in its ability to perform both traditional BO and LLM-based optimization, with both VI- and GP-based uncertainty quantification. It also stands out for its modular design and user-friendly interface.

5 EXPERIMENTS: A BENCHMARK OF COMBINING BUILDING BLOCKS.

Having introduced the machinery to modularly carry out the benchmark experiments on the molecular space, we benchmark the Bayesian graph- and language-based active learning strategies.

Data. While we are interested in benchmarking the ability of our active learning algorithms to optimize the *potency* of compounds, high-quality, consistent potency data in the public domain is scarce due to the close-source nature of most drug discovery campaigns. Nevertheless, from a methodological point of view, the potency function f is no more than a mapping from the graph structure to $\mathbb{R}$, and has the same *functional signature* as, and is dependent upon, the physiochemical properties of molecules. As such, we used the physical property datasets, ESOL [49], FreeSolv [50], and Lipophilicity from MoleculeNet [51]. The targets are normalized to be distributed within the range of $[0, 1]$, so the reported metrics are problem-agnostic [52]. Following the conceptual framework outlined in § 2, for all experiments, we start with an empty set, randomly select the first candidate, and iteratively refine the model based on the data already evaluated.

LLMs are not as performant as more traditional models. Despite the high variance in the data, we notice that the best models uniformly arise from the composition of GNN, GP, and EI (except for the last row for the DPO model to be introduced). The more LLMs are involved in the active



Marker	Model			Normalized Cumulative Regret		
	Rep.	U.Q.	Acq.	ESOL	FreeSolv	Lipophilicity
···	Random			2.27 ± 1.31	2.38 ± 1.57	2.19 ± 1.70
	GCN [6]	VI	EI	2.18 ± 1.15	1.62 ± 1.16	2.02 ± 1.40
			PoI	2.82 ± 1.58	2.18 ± 1.15	1.80 ± 0.78
		GP	EI	$\mathbf{1.61 \pm 0.67}$	1.67 ± 1.06	1.54 ± 1.07
			PoI	2.39 ± 0.38	2.52 ± 0.50	1.10 ± 0.43
	GAT [11]	VI	EI	1.29 ± 0.77	2.04 ± 1.29	2.65 ± 2.00
			PoI	2.41 ± 1.16	1.24 ± 0.52	1.60 ± 1.22
		GP	EI	2.08 ± 1.03	$\mathbf{1.08 \pm 0.46}$	1.73 ± 0.93
			PoI	2.17 ± 1.02	2.11 ± 0.57	$\mathbf{1.07 \pm 0.63}$
	`Llama-8B` [30]	VI	EI	2.32 ± 1.69	1.96 ± 1.36	1.91 ± 1.36
			PoI	2.88 ± 1.30	2.06 ± 1.40	1.84 ± 1.37
		GP	EI	2.13 ± 1.33	1.92 ± 1.19	1.81 ± 1.54
			PoI	2.25 ± 1.04	2.01 ± 1.09	1.09 ± 0.61
	`bert` [53]	GP	EI	2.10 ± 1.23	2.03 ± 1.30	1.90 ± 0.80
	`Llama-8B`		EI	2.40 ± 1.30	2.60 ± 1.45	1.95 ± 0.43
	`galactica-6.7b`			1.98 ± 0.85	1.70 ± 0.48	1.30 ± 0.85
	`Llama-8B`			2.77 ± 1.70	3.00 ± 1.24	2.17 ± 1.37
	$\texttt{Llama-8B}_{\text{description}}$			2.30 ± 1.82	1.95 ± 1.48	1.96 ± 1.18
	`galactica-6.7b` [54]			1.76 ± 1.07	2.04 ± 1.54	1.43 ± 1.23
	$\texttt{galactica-6.7b}_{\text{description}}$			2.15 ± 1.28	1.87 ± 1.44	1.89 ± 1.24
	`ChemLLM-7B` [20]			3.41 ± 1.16	2.66 ± 1.42	2.51 ± 1.48
	$\texttt{ChemLLM-7B}_{\text{description}}$			2.42 ± 1.39	2.86 ± 1.75	2.63 ± 1.86
	DPO (§ 6)			1.64 ± 1.24	$\mathbf{0.91 \pm 0.38}$	$\mathbf{1.03 \pm 0.45}$

Table 1: **Benchmark experiment on real-world dataset: The effect of replacing BO components with LLMs.** Normalized cumulative regret (↓) with various *representation*, *uncertainty quantification*, and *acquisition* modules. Representative configurations are also plotted: the maximum value y_{MAX} (averaged over 50 runs) plotted against the steps of acquisition. The random baseline is plotted in a dotted dark line. Confidence intervals are omitted in the figures for clarity.

learning pipeline, the worse the acquisition efficiency is—they can pass as feature extractor [38] when coupled with a GP (this is consistent with the findings of Ramos et al. [37]), achieving similar performance as a simple transformer [53]; when used as uncertainty quantification in an in-context manner, their performance deteriorates; and when used in an end-to-end manner to pick candidates directly, they act no different than the random acquisition function. At the same time, each graph-based, Bayesian model completes trials within a minute, whereas LLM-based models require by magnitude more time.

The effect of domain-specific models. There have been a plethora of LLMs for chemistry-related tasks [55], the most popular ones include Yu et al. [22]; Zhang et al. [20], which have been fine-tuned on instruction datasets [56]. Since they have "seen" more chemistry-related texts, it is reasonable to expect that they will perform better than the general-purpose models. This is the case for `galactica` [54], an LLM dedicated to science-related tasks, which outperforms the base `Llama` model.

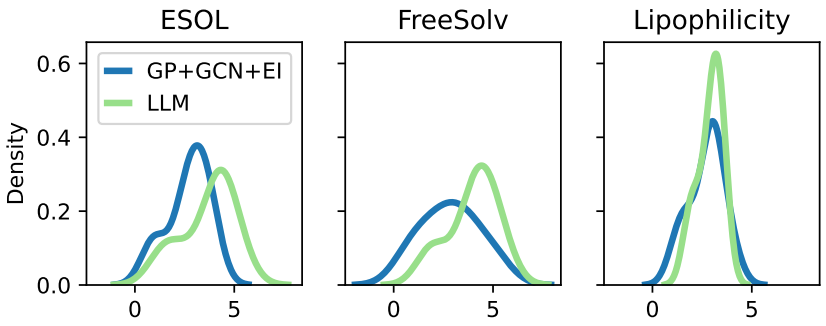


Figure 2: **Comparison of active learning algorithms on synthetic datasets.** Kernel density estimate (KDE) over the normalized cumulative regret (↓) with GCN + GP + EI and LLM (`Llama-8B`) acquisitions over a range of synthetic targets defined in Equation 14. The candidate space is taken from the real-world dataset.

Does knowing the target help? Again, the datasets employed in this section are reliably measured physiochemical properties. We decorate the prompt with the physical meaning of the targets in the trials marked with "description": aqueous solubility in ESOL, hydration free energies in FreeSolv, and the octanol/water distribution coefficient in Lipophilicity. Improvements are observed with descriptions added to the prompt.

Have LLMs already seen this data? A risk ubiquitous in benchmarking LLMs on real-world data in the public domain is that there is no guarantee that the data was not in the training set. For instance, the datasets benchmarked here are explicitly included in the training set of Yu et al. [22]. Despite this risk, Table 1 illustrates that even with possible leakage, LLMs is still less competitive compared to GNN + GP initialized afresh. For a fairer comparison nonetheless, we synthesize an artificial target:

$$f(\mathcal{G}) = \sum_i \lambda_i K_i(\mathcal{G}, g_i), \tag{14}$$

where K is a set of pre-defined graph kernels comparing the graph with certain fragments g_i, termed Morgan fingerprints [57], and λ_i is taken from a Gaussian distribution. With each set of $\{\lambda_i\}$, we can correspondingly define a target. As shown in Figure 2, with synthetic dataset, the difference between the traditional, graph- and GP-based acquisition and the LLM is even more obvious.

Experimental details and fixed degrees of freedom. Since the space of hyperparameter is vast and to find the best-performing model for each class is not in the scope of the paper, we prescribe a set of fixed experiment protocols for the graph-based Bayesian models benchmarked in Table 1. Within each round of acquisition, the neural network model is trained for 50 epochs with Adam [58] optimizer with learning rate $1e-3$ and L2 regularization factor $1e-5$; 8 samples are taken for all training and inference steps; and 1 attention head is used for GAT [11]. In this paper, we only consider exact GPs, leaving its variational counterpart [59] for future study. Even under these arbitrarily defined hyperparameters, these traditional models still perform better than more advanced LLMs.

6 Direct preference optimization (DPO) pretraining for more efficient optimization: To think like a GP, and better.

LLMs are routinely instruction-tuned [56] on supervised learning tasks to offer better predictions and "understandings" of certain candidate spaces. Without uncertainty calibration and carefully chosen acquisition functions, reliable predictions cannot be readily translated into efficient experimental design. In this section, we provide a general recipe to readily optimize the acquisition efficiency of LLMs in an end-to-end manner, i.e. taking over all of the representation, uncertainty quantification, and acquisition steps. We achieve this by learning from *all* of the choices (Equation 1) from successful trials driven by GNN + GP active learning algorithms.

Recall the sequential nature of the active learning—by assuming the greediness of Equation 1, we optimize the efficiency of the entire trial by optimizing that of each acquisition. We use direct preference optimization (DPO) [24] to achieve this goal. Specifically, note that in Table 1, the combination of GNN + GP is a uniformly strong baseline, we use this method to generate a cohort of N trials, and find the most efficient trial, judged by the lowest normalized cumulative regret. Based upon our hypothesis, each step in the trial is efficient and worth encouraging. We therefore pair each acquisition step with a step generated from the random acquisition function as *preferred* and *dispreferred* samples, arriving at the policy objective:

$$\mathcal{L}_{\texttt{DPO}} = -\mathbb{E}(\mathcal{G} \in \mathcal{P}, \mathcal{G}_t^{\mathcal{GP}} = \text{argmax}_{\mathcal{G}\in\mathcal{F}}\, \alpha_{\texttt{EI}}(p(y_i)|\mathcal{G}), P(\mathcal{G}^{\text{random}}) = \texttt{Categorical}(\mathcal{G}_{>t}))$$
$$[\log \sigma(\beta \log \frac{\pi_\theta(\mathcal{G}_t^{\mathcal{GP}}|\mathcal{G}_{<t})}{\pi_0(\mathcal{G}_t^{\mathcal{GP}}|\mathcal{G}_{<t})} - \beta \log \frac{\pi_\theta(\mathcal{G}_t^{\text{random}}|\mathcal{G}_{<t})}{\pi_0(\mathcal{G}_t^{\text{random}}|\mathcal{G}_{<t})})] \quad (15)$$

Descending this objective leads to the LLM active learner not only to think *like* a GNN + GP algorithm, but also generating only the most successful trials. As shown in the last row of Table 1, this turns out to be a highly useful strategy. Starting from a particularly small model of `Qwen-0.5B` [60], we tune the acquisition efficiency to surpass that of the strong baseline of GNN + GP. Even though this particular experiment is small and only demonstrative, this only positive result amidst the discouraging performance of LLMs offers a promising avenue for tuning LLMs to directly make step-wise decisions in an active learning setting.

7 Conclusion.

In this paper, we abstract drug discovery as a sequential active learning process, and survey the Bayesian optimization (BO) building blocks that can lead to principled and efficient experiment design. Next, we review possible ways for these building blocks to be replaced by LLMs. A software package, COLT, is provided for carrying out active learning experiments and assessing its efficiency. Using this package, we thoroughly benchmark the effects of replacing BO building blocks with LLMs, and found that such replacement causes only slowdown and performance drop. This suggests that a large quantity of noisy knowledge (represented by LLMs) is not as effective as a principled model with appropriate inductive biases (BO + GNN). Nevertheless, under the framework of direct preference optimization (DPO), we provide a recipe for tuning LLMs directly for generating step-wise optimization decisions.

References

[1] Steven M Paul, Daniel S Mytelka, Christopher T Dunwiddie, Charles C Persinger, Bernard H Munos, Stacy R Lindborg, and Aaron L Schacht. How to improve r&d productivity: the pharmaceutical industry's grand challenge. *Nature reviews Drug discovery*, 9(3):203–214, 2010.

[2] Michael Retchin, Yuanqing Wang, Kenichiro Takaba, and John D Chodera. Druggym: A testbed for the economics of autonomous drug discovery. *bioRxiv*, pages 2024–05, 2024.

[3] Júlia G.B. Pedreira, Lucas S. Franco, and Eliezer J. Barreiro. Chemical intuition in drug design and discovery. *Current Topics in Medicinal Chemistry*, 19(19):1679–1693, October 2019. doi: 10.2174/1568026619666190620144142. URL `https://doi.org/10.2174/1568026619666190620144142`.

[4] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. *CoRR*, abs/1703.02910, 2017. URL `http://arxiv.org/abs/1703.02910`.

[5] Peter I. Frazier. A tutorial on bayesian optimization, 2018.

[6] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016. URL `http://arxiv.org/abs/1609.02907`.

[7] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[8] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*, 2017.

[9] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in neural information processing systems*, pages 1024–1034, 2017.

[10] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[11] Petar Velikovi, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Li, and Yoshua Bengio. Graph attention networks, 2018.

[12] Yuanqing Wang and Kyunghyun Cho. Non-convolutional graph neural networks, 2024. URL https://arxiv.org/abs/2408.00165.

[13] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859877, April 2017. ISSN 1537-274X. doi: 10.1080/01621459.2017.1285773. URL http://dx.doi.org/10.1080/01621459.2017.1285773.

[14] Jacob R. Gardner, Geoff Pleiss, David Bindel, Kilian Q. Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with GPU acceleration. *CoRR*, abs/1809.11165, 2018. URL http://arxiv.org/abs/1809.11165.

[15] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[16] Markus Follmann, Hans Briem, Andreas Steinmeyer, Alexander Hillisch, Monika H Schmitt, Helmut Haning, and Heinrich Meier. An approach towards enhancement of a screening library: The next generation library initiative (ngli) at bayeragainst all odds? *Drug Discovery Today*, 24(3):668–672, 2019.

[17] Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, et al. Chembl: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 40(D1):D1100–D1107, 2012.

[18] Gengmo Zhou, Zhifeng Gao, Qiankun Ding, Hang Zheng, Hongteng Xu, Zhewei Wei, Linfeng Zhang, and Guolin Ke. Uni-mol: A universal 3d molecular representation learning framework. 2023.

[19] Ruoxi Sun, Hanjun Dai, and Adams Wei Yu. Does gnn pretraining help molecular representation?, 2022. URL https://arxiv.org/abs/2207.06010.

[20] Di Zhang, Wei Liu, Qian Tan, Jingdan Chen, Hang Yan, Yuliang Yan, Jiatong Li, Weiran Huang, Xiangyu Yue, Wanli Ouyang, Dongzhan Zhou, Shufei Zhang, Mao Su, Han-Sen Zhong, and Yuqiang Li. Chemllm: A chemical large language model, 2024. URL https://arxiv.org/abs/2402.06852.

[21] Nathan C Frey, Ryan Soklaski, Simon Axelrod, Siddharth Samsi, Rafael Gomez-Bombarelli, Connor W Coley, and Vijay Gadepally. Neural scaling of deep chemical models. *Nature Machine Intelligence*, 5(11):1297–1305, 2023.

[22] Botao Yu, Frazier N. Baker, Ziqi Chen, Xia Ning, and Huan Sun. Llasmol: Advancing large language models for chemistry with a large-scale, comprehensive, high-quality instruction tuning dataset, 2024. URL https://arxiv.org/abs/2402.09391.

[23] Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, pages 1–11, 2024.

[24] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

[25] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[26] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks, 2015.

[27] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Deep kernel learning, 2015. URL https://arxiv.org/abs/1511.02222.

[28] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms, 2012. URL https://arxiv.org/abs/1206.2944.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

[30] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothe Lacroix, Baptiste Rozire, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

[31] OpenAI. Chatgpt. https://chat.openai.com, 2023. Accessed: 2024-09-25.

[32] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

[33] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021.

[34] Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. Large language models for mathematical reasoning: Progresses and challenges, 2024. URL https://arxiv.org/abs/2402.00157.

[35] Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models : A critical investigation, 2023. URL https://arxiv.org/abs/2305.15771.

[36] Tennison Liu, Nicols Astorga, Nabeel Seedat, and Mihaela van der Schaar. Large language models to enhance bayesian optimization, 2024. URL https://arxiv.org/abs/2402.03921.

[37] Mayk Caldas Ramos, Shane S. Michtavy, Marc D. Porosoff, and Andrew D. White. Bayesian optimization of catalysts with in-context learning, 2023. URL https://arxiv.org/abs/2304.05341.

[38] Agustinus Kristiadi, Felix Strieth-Kalthoff, Marta Skreta, Pascal Poupart, Alan Aspuru-Guzik, and Geoff Pleiss. A sober look at LLMs for material discovery: Are they actually good for Bayesian optimization over molecules? In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 25603–25622. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/kristiadi24a.html.

[39] David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28 (1):31–36, 1988.

[40] Benedek Fabian, Thomas Edlich, Hlna Gaspar, Marwin Segler, Joshua Meyers, Marco Fiscato, and Mohamed Ahmed. Molecular representation learning with language models and domain-relevant auxiliary tasks, 2020. URL https://arxiv.org/abs/2011.13230.

[41] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. A survey on in-context learning, 2024. URL https://arxiv.org/abs/2301.00234.

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. URL https://arxiv.org/abs/1912.01703.

[43] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. Deep graph library: A graph-centric, highly-performant package for graph neural networks, 2020. URL https://arxiv.org/abs/1909.01315.

[44] Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. Pyro: Deep universal probabilistic programming, 2018. URL https://arxiv.org/abs/1810.09538.

[45] Andrew Gordon Wilson and Hannes Nickisch. Kernel interpolation for scalable structured gaussian processes (kiss-gp), 2015. URL https://arxiv.org/abs/1503.01057.

[46] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rmi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2020. URL https://arxiv.org/abs/1910.03771.

[47] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention, 2023. URL https://arxiv.org/abs/2309.06180.

[48] Ryan-Rhys Griffiths, Leo Klarner, Henry Moss, Aditya Ravuri, Sang Truong, Yuanqi Du, Samuel Stanton, Gary Tom, Bojana Rankovic, Arian Jamasb, Aryan Deshwal, Julius Schwartz, Austin Tripp, Gregory Kell, Simon Frieder, Anthony Bourached, Alex Chan, Jacob Moss, Chengzhi Guo, Johannes Peter Dürholt, Saudamini Chaurasia, Ji Won Park, Felix Strieth-Kalthoff, Alpha Lee, Bingqing Cheng, Alan Aspuru-Guzik, Philippe Schwaller, and Jian Tang. Gauche: A library for gaussian processes in chemistry. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 76923–76946. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/f2b1b2e974fa5ea622dd87f22815f423-Paper-Conference.pdf.

[49] John S Delaney. Esol: estimating aqueous solubility directly from molecular structure. *Journal of chemical information and computer sciences*, 44(3):1000–1005, 2004.

[50] David L Mobley and J Peter Guthrie. Freesolv: a database of experimental and calculated hydration free energies, with input files. *Journal of computer-aided molecular design*, 28: 711–720, 2014.

[51] Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. Moleculenet: A benchmark for molecular machine learning, 2018. URL https://arxiv.org/abs/1703.00564.

[52] Shali Jiang, Henry Chai, Javier Gonzalez, and Roman Garnett. Binoculars for efficient, nonmyopic sequential experimental design, 2020. URL https://arxiv.org/abs/1909.04568.

[53] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

[54] Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science, 2022. URL https://arxiv.org/abs/2211.09085.

[55] Kevin Maik Jablonka, Philippe Schwaller, Andres Ortega-Guerrero, and Berend Smit. Is gpt-3 all you need for low-data discovery in chemistry? 2023.

[56] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. Instruction tuning for large language models: A survey, 2024. URL https://arxiv.org/abs/2308.10792.

[57] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

[58] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

[59] Dustin Tran, Rajesh Ranganath, and David M. Blei. The variational gaussian process, 2016. URL https://arxiv.org/abs/1511.06499.

[60] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report, 2023. URL https://arxiv.org/abs/2309.16609.