Position: Leverage Foundational Models for Black-Box Optimization

Xingyou Song [1] **Yingtao Tian** [1] **Robert Tjarko Lange** [2] **Chansoo Lee** [1] **Yujin Tang** [2] **Yutian Chen** [1]

Abstract

Undeniably, Large Language Models (LLMs) have stirred an extraordinary wave of innovation in the machine learning research domain, resulting in substantial impact across diverse fields such as reinforcement learning, robotics, and computer vision. Their incorporation has been rapid and transformative, marking a significant paradigm shift in the field of machine learning research. However, the field of experimental design, grounded on black-box optimization, has been much less affected by such a paradigm shift, even though integrating LLMs with optimization presents a unique landscape ripe for exploration. In this position paper, we frame the field of black-box optimization around sequence-based foundation models and organize their relationship with previous literature. We discuss the most promising ways foundational language models can revolutionize optimization, which include harnessing the vast wealth of information encapsulated in free-form text to enrich task comprehension, utilizing highly flexible sequence models such as Transformers to engineer superior optimization strategies, and enhancing performance prediction over previously unseen search spaces.

1. Introduction

Black-box optimization (BBO) refers to a class of techniques which use minimally observed information to maximize an objective function. Also known as *derivative-free* optimization or *zeroth-order* optimization, the only feedback for an optimizer is the objective value at a given query point, in the absence of additional information such as gradients and second-order derivatives. BBO is broadly prevalent across domains involving experimental design where computing gradients is impossible or infeasible, including automated machine learning (Feurer et al., 2015; Real et al., 2020; Mellor et al., 2021), drug discovery (Turner et al., 2021), and biological/chemical design (Angermueller et al., 2020). As an instructive example, in order to improve the performance for a classification task, one may tune a neural network's architecture. Since the accuracy is typically not differentiable with respect to hyperparameters that express design decisions such number of layers, as hyperparameters are varied, one must repeatedly train expensive models over multiple *trials*, to obtain a single accuracy metric. In order to minimize expensive costs, the core challenge in BBO is to **efficiently** search for parameters which maximize the objective.

Traditional black-box algorithms such as random search (Droste et al., 2006), evolutionary methods (Hansen, 2016), and Bayesian Optimization (Shahriari et al., 2016) have been developed robustly to optimize a wide range of black-box objective functions. Ironically, despite the term "black-box", many algorithms inherently perform better based on accurate assumptions about the nature of the objective function. These assumptions, or *priors*, significantly affect the algorithm's key behaviors, such as its predictions over the objective landscape using past observations, and its ability to balance exploration and exploitation to propose the next query point.

Manually designing these useful priors is difficult and is exacerbated when the prior may not match real world objectives or the prior for one category of tasks may not easily generalize to others, leading to high costs of design. One possible solution is that one may *learn* the prior if provided with realistic objective evaluations, leading to efficient BBO algorithms without needing to explicitly specify the prior. For instance, if given previous evaluations from multiple neural network optimization tasks, one may learn a specific algorithm to better tune the learning rate than a Bayesian optimization algorithm designed more generally for optimizing smooth functions. This type of data-driven approach is known as *learning to optimize* (Li & Malik, 2016; Chen et al., 2017; 2022a), a sub-area of meta-learning.

While there have been numerous emergent works based on learning to optimize, the optimization community has not seen wide adoption of these techniques, due to several common obstacles. To list a few:

[1]Google DeepMind [2]Sakana AI. Correspondence to: Xingyou Song <xingyousong@google.com>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

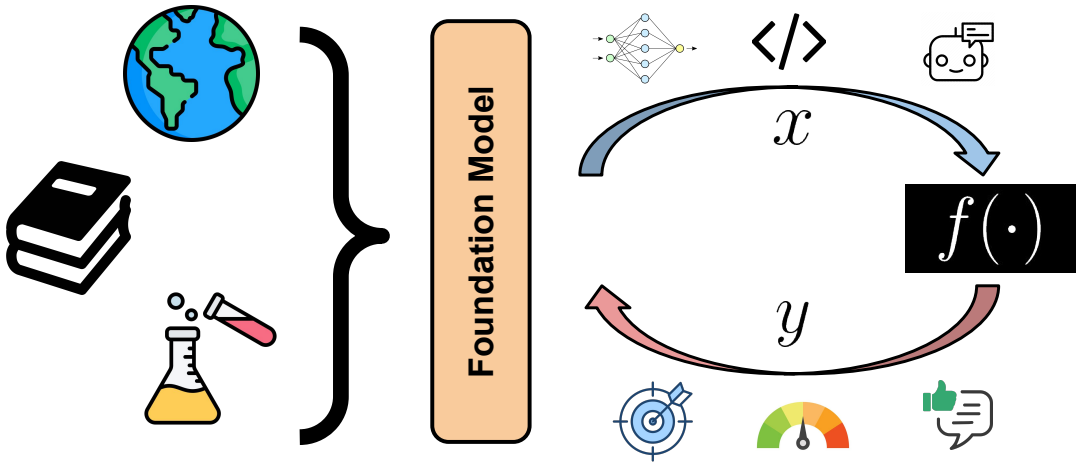

Figure 1. Foundation Models can learn priors from a wide variety of sources, such as world knowledge, domain-specific documents, and actual experimental evaluations. Such models can then perform black-box optimization over various search spaces (e.g. hyperparameters, code, natural language) and feedbacks (numeric values, categorical ratings, and subjective sentiment.

1. **Lack of generalizability beyond meta-training.** Meta-overfitting (Rajendran et al., 2020) often prevents learned BBO algorithms from generalizing to test functions, which may have different properties and conditions from ones seen during meta-training, such as smoothness, search space dimensionality, and length of optimization.
2. **Low flexibility in exploiting information from data.** BBO is often formulated as a numeric optimization problem, but BBO tasks in practice often contain rich unstructured information beyond numbers, such as task description, search space configuration, historic experiments, textual feedback, and even additional measurements such as learning curves.
3. **Little scalability for diverse optimization tasks.** Most learned BBO algorithms are designed for specific types of tasks. A specific algorithm has limited model capacity in handling data from a diverse set of tasks and is typically restricted to a fixed search space. One must train a completely new optimizer for a new optimization problem family, and the fixed search space limitation further limits the training data one could obtain, which exacerbates the scalability issue.

The recent rise of foundation models (Bommasani et al., 2021) via Transformers (Vaswani et al., 2017) and their use in Large Language Models (LLMs) have stirred an extraordinary wave of innovation in various machine learning domains such as natural language, programming, robotics, and mathematical reasoning. Incorporation has been rapid and transformative, marking a significant paradigm shift where foundation models can be trained on broad data at scale and adapted to a wide range of tasks. In contrast, such methods have not been thoroughly studied in the area of BBO, even though applicable training data exist, consisting of not only optimization trajectories but also relevant world knowledge on experimental design and optimization. Such knowledge could adapted to BBO tasks over various search spaces and data types, as illustrated in Figure 1.

This position paper advocates for wider research and adoption of Transformers and LLMs for black-box optimization, as they possess a few key benefits which are desirable to address the aforementioned challenges in learned BBO:

1. The Transformer's input format allows **modeling a wide variety of data**, ranging from fixed dimensional vectors to sequences of text and even multi-modal data, given the proper encoding schemes. This input format allows the possibility of learning a single model from mixed datasets of diverse optimization tasks, each with task-specific side information.
2. The **Transformer's superior scalability** is a much needed property for learning a general BBO algorithm from large datasets compared to traditional machine learning models.
3. The Transformer's **in-context learning capacity** is useful for improving a learned algorithm's generalizability to new settings.

Our paper is structured as follows: Section 2 provides preliminaries and notation on BBO, and Section 3 conducts thorough survey of previous works, organized by approaches with gradually increasing generalities and relationship with sequence-based models, ultimately towards LLM-

based techniques. In Section 4, we then identify challenges and advocate for techniques which are crucial in advancing the field of learned optimizers. These include data-driven training, better data representations and multi-modality, and more flexible model prompting, with additional emphasis on improved BBO benchmarking. Lastly, in Section 5, we provide a vision for next generation LLM-based optimizers, which are able to integrate multi-modal data, incorporate user-provided feedback, and manage problem-specific information over long contexts.

2. Preliminaries and Notation

Problem 1 Experimental Design Problem

Require: Problem meta-description $m \in \mathcal{M}$, search space $\mathcal{X}$, and feedback function $f \in \mathcal{F}$, Algorithm $\mathcal{A}$
  Initialize $\mathcal{A}$ using all provided information.
  Initialize history $h \leftarrow []$
  for t=1, 2, ... **until** end condition is met **do**
    Generate a suggestion: $x_t \leftarrow \mathcal{A}(m, h_{1:t-1})$
    Receive feedback: $y_t \leftarrow f(x_t)$
    Update history: $h \leftarrow$ concatenate(h, x_t, y_t).
  end for
  Return $h_{1:t}$ **as** H

We begin by defining the generic problem of experimental design. At the t-th iteration, the algorithm $\mathcal{A}$ suggests x_t in the search space $\mathcal{X}$ and receives feedback $y_t = f(x_t)$, from a feedback space $\mathcal{Y}$. We use $h_{s:t}$ to denote the optimization *history* from the s-th iteration up to the t-th iteration, i.e. the sequence of suggestions and feedbacks $(x_s, y_s, x_{s+1}, y_{s+1}, \ldots, x_t, y_t)$. We may omit the subscript for h if it is clear from the context. We use $\mathcal{H}$ to denote the space of all possible trajectories of any length, and uppercase H without the subscript to denote the full trajectory of the run.

Note that what makes our problem formulation different from the standard black-box optimization problem is the existence *problem meta-description*. This meta-description m contains some information that may hint towards a more specific subset of the function set $\mathcal{F}$ which f was chosen from.

In this paper, we often consider the setting where we have access to a database of historic runs, where each element is a tuple $(m_i, \mathcal{X}_i, H_i, \mathcal{A}_i)$; note that the feedback function f_i itself is *not* included.

2.1. Search Spaces as Sequences

To better understand our motivation for using foundation models, one important insight is to connect search spaces to the notion of *formal grammars* (Hopcroft & Ullman, 1979). Without introducing overly technical background on grammars, it suffices to see that $\mathcal{X}$ is representable as a sequence of atomic *parameter configurations*, or alphabets $\mathcal{X}^{(1)}, \mathcal{X}^{(2)}, \ldots$ and thus each x can be seen as a collection of parameters, or a "string" $(x^{(1)}, x^{(2)}, \ldots)$. In addition, there may be a *feasibility function* $\phi : \mathcal{X}^{(1)} \times \mathcal{X}^{(2)} \times \ldots \to \{0, 1\}$ which determines whether x is admissible in the search space.

Note that while a search space $\mathcal{X}$ can be commonly reparameterized into a new space $\mathcal{X}'$, certain fundamental invariants remain which dictate the inherent properties of the space. These are:

- *Parameter Type:* $\mathcal{X}^{(i)}$ can be continuous (e.g. subinterval of $\mathbb{R}$) or discrete (e.g. $\mathcal{X}^{(i)} \subseteq \{1, 2, \ldots, n\}$).
- *Constraints:* $\mathcal{X}$ can be unconstrained or may possess a nontrivial ϕ.
- *Length Boundedness:* x may be of unbounded length $(x^{(1)}, x^{(2)}, \ldots)$ or explicitly bounded (both below and above).

The particularly nuanced invariant is ϕ, which is important to organize into the following categories:

- **Unconstrained:** $\phi(x) = 1$ always.
- **Inductive:** ϕ can be factored locally as $\prod_{j=0} \phi(x^{(j+1)}|x^{(1:j)})$ where $\phi(x^{(j+1)}|x^{(1:j)})$ is efficiently computable and can be seen as a one-step restrictor mapping $x^{(1:j)} = (x^{(1)}, \ldots x^{(j)})$ to a subset of allowed values in $\mathcal{X}^{(j+1)}$.
- **Multi-step Inductive:** The above factorization requires use of k-step restrictors $\phi(x^{(j+1:j+k)}|x^{(1:j)})$ where $k \geq 2$ is reasonably small.
- **Global:** If ϕ does not admit an efficiently computable factorization, then determining feasibility only occurs once the entire x is formed, and ϕ can be seen as a global "compilation check".

3. Previous Works and Motivation

3.1. Previous Applications

While it is natural to organize optimization problems by application domain, it is far more useful for researchers to organize problems by their fundamental search spaces and invariants defined in Section 2.1, which heavily influence algorithm design. We organize these problems in Table 1.

By organizing using ϕ, we can see that traditional BBO problems such as continuous optimization (Elhara et al., 2019) and protein sequence design (Angermueller et al., 2020) have consisted of unconstrained Cartesian spaces. In contrast, combinatorial problems such as Traveling Salesman (Flood, 1956) and other graph problems (Balakrishnan

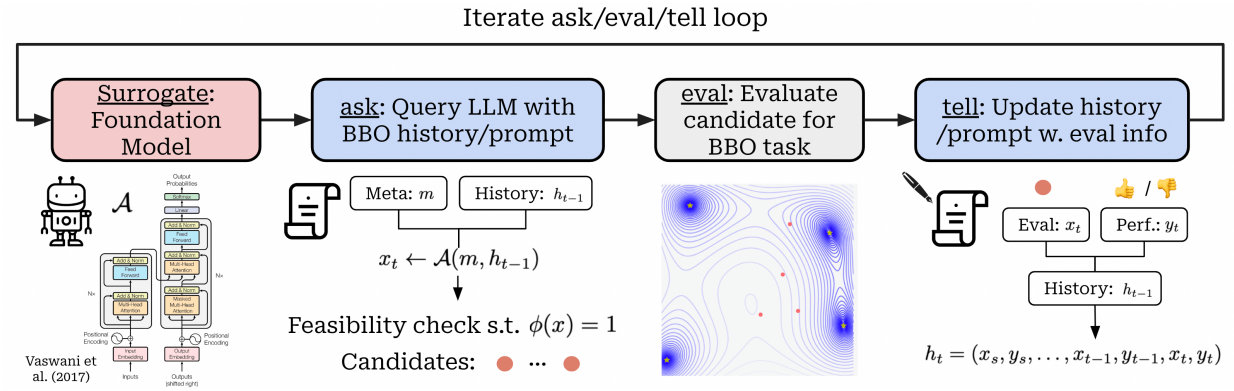

Figure 2. Black-box optimization loop with sequential foundation models. Using metadata m and history h, the model proposes candidates x which are checked for feasibility, evaluated, and then appended to the history.

Application \ Invariant	Parameters	Constraints	Length Bound
Traditional	Any	Any	Bounded
Combinatorial	Discrete	Inductive	Bounded
Genetic Programming	Any	Multi-step Inductive	Any
Free-form Code	Discrete	Global	Unbounded
Free-form Prompt	Discrete	None	Unbounded

Table 1. Example applications categorized by search space invariants.

& Ranganathan, 2012) use primitives such as permutations and combinations, which are inductive constraints, as the next possible values for $x^{(j+1)}$ are determined by previously chosen $x^{(1:j)} \subset \{1, \ldots, n\}$.

In the domain of program search, classic genetic programming (Real et al., 2020) factorizes the search space by smaller subtrees of possible symbols, and thus use multi-step inductive constraints. However, free-form code search (Romera-Paredes et al., 2023) is constrained by both compilation and runtime errors, which can only be checked after an entire program is globally created. Ignoring optional grammar constraints, ultimately the field of prompt optimization defines objectives over arbitrary strings.

3.2. Previous Techniques

From our sequential formalization of both the experimental design problem and their underlying search spaces, it is thus natural to see the relevance of models which use sequential representations, particularly for more complex applications. We outline key previous works in order of development, which have crucially developed along this direction, with their capabilities shown in Table 2, eventually leading towards the use of learned foundation models in BBO loops, as illustrated in Figure 2.

Non-learnable: These consist of purely hand-designed rules for proposing x_{t+1} based on $h_{1:t}$ and can be deterministic or stochastic. A common theme in the evolutionary algorithm literature is around the idea of *mutation*, in which only a fixed-size pool of historical trials are used to construct x_{t+1} by changing every parameter in $(x_{t+1}^{(1)}, x_{t+1}^{(2)}, ...)$ using perturbations and crossover. Such methods are usually sample inefficient, as many algorithmic components are based on unguided randomness.

Feature-based: Traditional model-driven BBO requires formatting (x, y) as *features*, in particular creating representation mappings $\mathcal{X} \to \mathbb{R}^d$ to allow the construction of learnable regressors $\mathbb{R}^d \to \mathbb{R}$ such as Tree-structured Parzen Estimator (TPE) (Bergstra et al., 2011), Random Forests (Hutter et al., 2011), Gaussian Processes (GPs) (Snoek et al., 2012), and feed-forward neural networks to guide search. While feature construction is straightforward for Cartesian spaces, it is considerably more complex for conditional spaces such as combinatorics (Oh et al., 2019) or graph spaces (Ru et al., 2021).

Generally, feature-based methods have allowed the algorithm to adapt to incoming observations within the current task, although their performance is bounded by the limited number of available observations and their hand-designed prior.

Meta-learned: In order to take advantage of information from observations of related tasks, transfer-learning / meta-learning algorithms are proposed to learn task-independent prior and optional task-specific parameters. Representative feature-based meta-learned methods often combine Gaussian Processes with deep-learning kernels (Swersky et al., 2013; Yogatama & Mann, 2014; Perrone et al., 2018; Volpp et al., 2019; Wistuba & Grabocka, 2021; Wang et al., 2021). While more specific prior can be learned in this approach,

Method \ Capabilities	In-task Adaptability	Across-Task Transferrable	In-context Learning	Search Space Transferrable	Length Scalable	Multi-modality	Built-in NLP
Non-learnable	✗	✗	✗	✗	✗	✗	✗
Feature-based	✓	✗	✗	✗	✗	✗	✗
Meta-learned	✓	✓	✗	✗	✗	✗	✗
Sequential History	✓	✓	✓	✗	✗	✗	✗
Sequential Search Space	✓	✓	✓	✓	✗	✗	✗
Attention-based	✓	✓	✓	✓	✓	✗	✗
Token-based	✓	✓	✓	✓	✓	✓	✗
LLMs	✓	✓	✓	✓	✓	✓	✓

Table 2. Classes of methods organized by their capabilities. Note: Method names based on increasing development order - e.g. "Attention-based" can consist of techniques up to their development such as meta-learning, but not LLMs.

in-task finetuning is still required for optimal performance (Wistuba & Grabocka, 2021).

Sequential History: In order to utilize a static model which can explicitly process H and directly output a x, a sequential model such as a recurrent neural network (RNN) introduces a weight-indepdendent sequence axis for additional *in-context learning*. For the multi-turn bandit case, (Chen et al., 2017) places the historical trajectory H on the sequence axis to propose a new x. Thus the model's weights are independent of the length of H and can arbitrarily process any history.

Sequential Search Space: Sequential models are also used to process search spaces with an inductive ϕ. (Bello et al., 2017) uses an RNN's length axis along with masking based on ϕ, to construct a distribution $p_\theta(x) \in \mathcal{P}(\mathcal{X})$ over a family of combinatorial spaces. This can be seen as a form of restricted decoding over the parameters $(x^{(1)}, x^{(2)}, ...)$ of x. By placing the parameters x along the length axis, the model's weights are thus independent of the size of x and the search space.

Attention-based: Since RNNs have difficulty scaling to accommodate long sequences, the attention mechanism has been adopted by e.g. Lange et al. (2023b;a) to construct evolutionary and genetic algorithms possessing strong generalization capabilities. In addition, (Müller et al., 2023; 2021; Nguyen & Grover, 2022) train Transformers to conduct in-context Bayesian inference as a surrogate model for Bayesian optimization, while (Nguyen et al., 2023; Mashkaria et al., 2023) use Transformers to directly propose suggestions x.

Token-based: In order to obtain weight-independence for both the history *and* search space, the OptFormer (Chen et al., 2022c) represents every parameter value as a *token* to avoid constructing features altogether. Metadata was additionally tokenized and processed as text. Despite the use of text as input, the model was trained on a relatively small text corpus and did not demonstrate the emergent capacity to understand semantic meanings of metadata yet. (Lange et al., 2024a) alternatively learns its own embedding to map each $x^{(i)}$ to a single vector.

While there have yet to be optimizers which process other modes of data such as images and audio, token-based methods allow the use of unified vocabularies which can represent multiple modes simultaneously, which may be of interest in the future for representing more exotic forms of m, x, y.

Natural Language: LLMs provide a powerful general-purpose tool for leveraging additional text-based BBO information. There exist several applications using off-the-shelf pre-trained models as mutation operators in the context of code-based settings. More specifically, (Lehman et al., 2023) embedded LLMs into Genetic Programming, i.e. the model serves as a mutation operator optimizing morphologies and behaviors on the code level. (Chen et al., 2023; Nasir et al., 2023) use an LLM to adaptively mutate and cross-over code for evolutionary Neural Architecture Search. Furthermore, (Meyerson et al., 2023) introduce Language Model Crossover where they concatenate parent solutions into a prompt and collect offsprings to evaluate from the text-based output. Here, the LLM has to act as a variation operator and thereby evolves genomes representable as text strings (Liu et al., 2023b).

While these approaches were mostly applied to fairly narrow use-cases, more recent works have been leveraging LLMs as general-purpose optimizers. (Yang et al., 2023) first leveraged LLMs as pure black-box optimizers. More specifically, they showed that PaLM-2 (Anil et al., 2023)

models to directly output candidates and perform linear regression, prompt optimization, and other tasks. LLMs can also be used to propose distributions for sampling x's, as shown for single-objective (Lange et al., 2024b; Liu et al., 2023e) and multi-objective (Liu et al., 2023a) evolutionary optimization. (Zhang et al., 2023; Nie et al., 2023) further showed that LLMs can be applied to hyperparameter optimization problems using natural language instructions and feedback. Additionally, LLMs can be used for related subtasks such as multi-task regression (Song et al., 2024) and surrogate modeling (Liu et al., 2024).

4. Challenges and Required Techniques

Despite the significant potential benefits of learned optimizers, so far they have not been widely used, let alone adopted in production-grade optimization systems (Golovin et al., 2017; Facebook, 2021). We believe this is due to a variety of key technical challenges, many of which can be resolved under the current status-quo of using Transformer-based foundation models, while other challenges will require further improvements, summarized in Figure 3.

Data Representation and Multimodality: Most meta-learning based BBO methods consider a fixed search space $\mathcal{X}$ and feedback space $\mathcal{Y}$ (Swersky et al., 2013; Wistuba & Grabocka, 2021; Chen et al., 2017). Specific embedding methods must be manually crafted for each domain (hyperparameters, graphs, code, etc.) This drastically limits the scope in which a learned method can be transferred over. As a result, an optimizer must be learned from scratch for every new type of application or even search-space dimension.

In contrast, sequence-based and more broadly text-based representations, when consumed by Transformer models and applied over both the historical and search space axes of optimization problems, greatly increase generality and transferrability. More broadly is the ability to incorporate multimodal knowledge, outside of using textual representations. For instance, one may be able to predict the outcome of a machine learning experiment not only from observing hyperparameters, but also the code used for training, neural network parameters (Unterthiner et al., 2020), and even the dataset used. Another example in a wet-lab scenario could be to express (m, x, y) as images, which can be more expressive than pure text or numbers. While modality-specific encoders for BBO have yet to be applied, general usage over other domains has been widely studied and achieved, such as across-domain transfer (Raffel et al., 2020; Brown et al., 2020) and across-modality alignment (Radford et al., 2021; Girdhar et al., 2023).

Open-questions remain on the optimal tokenization of optimization-specific domains, especially for numeric and mathematical objects. Currently, general consensus from previous literature (Chen et al., 2022b) suggests that numeric objects (e.g. floats, graphs) should at least be tokenized according to their unit building blocks (e.g. digits, vertices). Thus it is not clear that representing numbers in standard human-readable format is optimal, especially as certain tokenizers (Kudo & Richardson, 2018) will not split such numbers digit-by-digit — for example `123.4` may be split into tokens representing `[12, 3., 4]`.

Interfaces: One would expect that due to common fundamental ideas behind LLMs, their user interface should also remain consistent. This is not true in practice however, as multiple different LLMs provide APIs with varying degrees of interactivity. For instance, open-source LLMs (Touvron et al., 2023) allow the user to fine-tune the model against custom data, while closed-source LLMs (OpenAI, 2023) only provide remote procedure calls. Furthermore, many closed-source services utilize different technical choices when defining an "embedding" or when performing inference and decoding. While there has been some recent effort for creating unified APIs (Cheng et al., 2023; Pham et al., 2023), these have mostly been for the purpose of prompting.

For optimization in particular, currently there is no unifying format which the community agrees upon. This is further constrained by a strong need to *deserialize* string outputs back into x and y. While LLMs sufficiently trained on code data can output tabular JSON formats, e.g. {`batch_size:128,learning_rate:0.1`} used in hyperparameter optimization, following more sophisticated combinatorial and numeric constraints is still an open question.

There is additional interest in functionality beyond regular x-proposal or y-regression for BBO. While the LLM community has actively studied interpretability (Luo & Specia, 2024), attribution (Li et al., 2023b), and uncertainty quantification (Xiong et al., 2023) for general LLM assistant scenarios, there has been little work in the context of BBO and decision-making in bandit settings. Such BBO-specific functionalities are likely to be more difficult to induce, as they require advanced mathematical and numerical reasoning beyond current LLM capabilities.

Training Datasets: Learned BBO algorithms are significantly dependent on both the quality and quantity of data available. The most obvious useful data are function evaluations (x, y), as they can be used to form a prior on the nature of the function being optimized. The most common form of usage are for pretraining regressors which can then guide the search process. Due to limitations to fixed search spaces, traditional methods do not have or consider large real-world training datasets. As a result, there is currently a lack of large-scale open-source evaluation datasets. This consequently limits a learned BBO method's generality, even though Chen et al. (2022c) is the first work which has shown

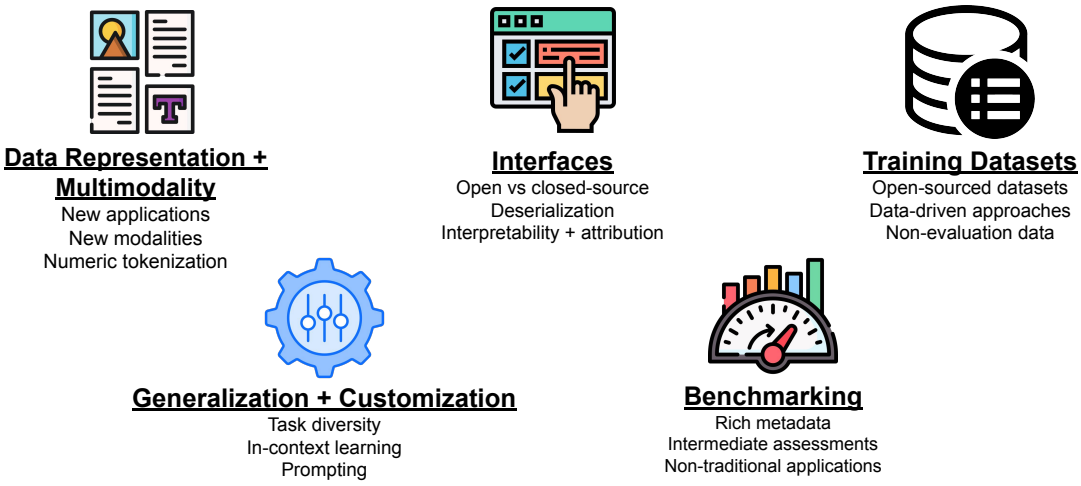

Figure 3. Summary of future challenges and open questions for BBO with LLMs.

that training a foundation model over large-scale hyperparameter tuning data collected by Google Vizier (Golovin et al., 2017) leads to robust generalization to new tuning tasks with different search and feedback spaces.

Unfortunately, industry datasets such as Google Vizier (Golovin et al., 2017) and Ax (Facebook, 2021) with rich task-specific metadata, are unlikely to be fully open-sourced due to proprietary legal and privacy concerns. While there have been some efforts to standardize smaller-sized public datasets (Eggensperger et al., 2021; Trabucco et al., 2022) and centralize community-driven data over an open platform such as OpenML (Bischl et al., 2021), an important question is when the BBO community may be able to truly embrace data-driven approaches.

A further open-question remains on how to obtain more domain knowledge besides using function evaluations. For instance, techniques from protein design (Castro et al., 2022; Madani et al., 2023) train foundation models over protein sequence data as a form of knowledge absorption. It remains to be seen whether additional "optimization knowledge" can be gained from free-form datasets such as textbooks, academic papers, and whether these can lead to emergent behaviors via generative pretraining and RL-finetuning. For example, the field of hyperparameter optimization could be improved by semantically understanding the nature of the tuned objective - e.g. academic knowledge of the term "batch size" could lead to better hyperparameter proposals.

Generalization and Customization: The primary challenge of meta-learning is its limited generalization to new tasks from different domains, search space dimensions, constraints, or simply different trajectory lengths. This is mostly due to the issue of *meta-overfitting* where the optimizer has overfitted to a limited number of optimization tasks with low diversity. Expanding the training set of tasks and optimizer model size could be effective solutions as suggested by LLM scaling laws. Additionally, the impressive in-context learning capacity (Brown et al., 2020) of a large Transformer allows a pretrained model to generalize to a new setting with few demonstrations without changing its weights. This would be an intriguing feature for adapting meta-learned BBO methods to new application tasks as it is typically difficult to obtain a large number of similar optimization tasks for adaptation in real applications.

Related to generalization is the customization of a pretrained optimizer to different use cases. It is necessary that a model understands the user's *intent* to optimize a function. For example, a user may prefer more exploration in a preliminary experimental design phase with a small budget while another user prefers to find a good solution as soon as possible. Furthermore, different users may require different safety / exploration constraints and provide various side information about the objective. Training a new optimizer for every use case is not feasible in practice. In contrast, the sequential input format of LLMs permits efficiently customizing its behavior through prompting, which can provide additional capabilities such as providing explanations for why the model suggested x or confidence over predictions of y-values.

While previous works (Chen et al., 2021; 2022c; Lange et al., 2024a;b) have used Transformers and even LLMs as the underlying model, they still only *implicitly* learn the intent for improvement by manipulating the sequential order of trials, such as least-to-most sorting by y-values and inversely using a query y to obtain a response x. The next generation of optimizers must allow *explicit* commands for optimization-related behaviors.

Benchmarking: Currently, most of BBO benchmarking requires well-defined, simple search spaces and objectives

with minimal contexts, to follow the namesake "black-box" and provide fair comparisons between general-purpose methods. These include extensive and official benchmarking tasks such as BBOB (Elhara et al., 2019), NASBench (Ying et al., 2019) or NeuroEvoBench (Lange et al., 2023c). However, we advocate for a wider spectrum of benchmarks that do not follow this regime, as they are crucial to the future of BBO.

For starters, there is a need for **metadata-rich black-box optimization**, especially as real-world problems inherently are not truly black-box. These benchmarks should be designed to emphasize an optimizer's utilization of rich metadata. In particular, this alludes to the use of additional language encoders in optimizers not only limited to LLMs, such as (Lange et al., 2024b; Chen et al., 2022c) which have shown promise over in real-world problems by using dedicated decoders.

As systems become more integrated with LLMs end-to-end, there is also a need to assess intermediate decision-making capabilities which are not typically measured by opaque traditional BBO metrics such as regret or best-objective. As shown recently via multi-armed bandit problems, (Krishnamurthy et al., 2024) claims LLMs such as GPT-4 to be quite poor at balancing explore-exploit tradeoffs. Meanwhile, (Song et al., 2024) demonstrated strong results on regression for fine-tuned language models. Another important capability is constraint-following, not just over mathematical objects but also ambiguous natural language constraints (e.g. "Give a batch size which fits into GPU memory").

Furthermore, there are important yet non-obvious applications that can be framed as black-box optimization, particularly over more exotic search spaces or feedback formats. For example, there has been recent interest in evaluating and benchmarking LLM-based code generation (Ji et al., 2023; Fu et al., 2023; Xu et al., 2022). Human-based reward modeling and reinforcement learning from human feedback (RLHF) can also respectively be seen as regression and black-box surrogate optimization (Ramamurthy et al., 2022; Wang et al., 2018). In addition, heavily-used techniques such as prompt engineering possess domain-specific guidelines (Wang et al., 2023b;a) but still lack well-accepted benchmarks and evaluation paradigms. Ultimately it remains an open problem to integrate these applications into a setting comparable to that of traditional BBO problems.

5. Future Directions

We identified and discussed challenges hindering progress in applying LLM and Transformer-based models specifically over optimization tasks, as well as their existing solutions in Section 4. These insights highlight key areas where focused research efforts can lead to substantial advancements. Venturing further into a domain that permits a more extended time horizon and a wider scope reveals intriguing and promising directions for future research.

The envisioning of a universal LLM, adept in both natural language understanding and executing complex optimization tasks, marks a significant leap forward in AI technology. Such a forward-looking model, with its capacity to **process user-provided metadata about target optimization problems** through straightforward instructions, promises transformative impact across numerous sectors. Imagine its application in enhancing human-robot interaction (Brohan et al., 2023; Yu et al., 2023), revolutionizing autonomous driving systems (Sha et al., 2023), innovating reward design in learning environments (Ma et al., 2023), and redefining efficiency in logistics planning (Li et al., 2023a). Moreover, the scope of this model's capabilities extends well beyond these areas. Envision, for instance, its application in a multi-agent setting where, through self-dialogue, the model could autonomously uncover and propose solutions to complex problems that elude human detection. Such innovative capabilities hint at a promising trajectory towards the realization of artificial general intelligence.

Realizing the vision of a universal LLM for complex optimization tasks hinges on overcoming significant hurdles, but recent promising strides in related areas signal a path forward. One of the primary challenges is **managing long context lengths in problem descriptions**, a task where methods like those in (Jin et al., 2024; Xiao et al., 2023) show promise, yet their direct application to optimization-focused LLMs remains to be validated. Equally important is the **integration of multi-modal data**, crucial for a holistic understanding of complex problems, with current techniques in (Liu et al., 2023d;c) potentially serving as a foundation. Furthermore, the idea of model composition, perhaps involving a network of specialized LLMs as indicated by research in (Shen et al., 2023; Bansal et al., 2024), opens new avenues but also demands further exploration. Crucially, these challenges underscore the need for enhanced collaboration and open research, urging a collective approach across different LLMs and disciplines to unlock the full potential of such models.

Acknowledgements

We thank Daniel Golovin and Sagi Perel for helpful feedback during the drafting of this manuscript. We further thank Aviral Kumar, Bert Chan, Zi Wang, Frank Hutter, and the AutoML Conference community for early discussions, and Heiga Zen for continuing support. We finally thank anonymous reviewers for their valuable feedback.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Angermueller, C., Belanger, D., Gane, A., Mariet, Z., Dohan, D., Murphy, K., Colwell, L., and Sculley, D. Population-based black-box optimization for biological sequence design. In *International Conference on Machine Learning*, pp. 324–334. PMLR, 2020.

Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Balakrishnan, R. and Ranganathan, K. *A textbook of graph theory*. Springer Science & Business Media, 2012.

Bansal, R., Samanta, B., Dalmia, S., Gupta, N., Vashishth, S., Ganapathy, S., Bapna, A., Jain, P., and Talukdar, P. Llm augmented llms: Expanding capabilities through composition. *arXiv preprint arXiv:2401.02412*, 2024.

Bello, I., Pham, H., Le, Q. V., Norouzi, M., and Bengio, S. Neural combinatorial optimization with reinforcement learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. Algorithms for hyper-parameter optimization. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

Bischl, B., Casalicchio, G., Feurer, M., Gijsbers, P., Hutter, F., Lang, M., Mantovani, R. G., van Rijn, J. N., and Vanschoren, J. Openml: A benchmarking layer on top of openml to quickly create, download, and share systematic benchmarks. *NeurIPS*, 2021.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R. B., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N. S., Chen, A. S., Creel, K., Davis, J. Q., Demszky, D., Donahue, C., Doumbouya, M., Durmus, E., Ermon, S., Etchemendy, J., Ethayarajh, K., Fei-Fei, L., Finn, C., Gale, T., Gillespie, L., Goel, K., Goodman, N. D., Grossman, S., Guha, N., Hashimoto, T., Henderson, P., Hewitt, J., Ho, D. E., Hong, J., Hsu, K., Huang, J., Icard, T., Jain, S., Jurafsky, D., Kalluri, P., Karamcheti, S., Keeling, G., Khani, F., Khattab, O., Koh, P. W., Krass, M. S., Krishna, R., Kuditipudi, R., and et al. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258, 2021.

Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Chen, X., Choromanski, K., Ding, T., Driess, D., Dubey, A., Finn, C., et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Castro, E., Godavarthi, A., Rubinfien, J., Givechian, K. B., Bhaskar, D., and Krishnaswamy, S. Transformer-based protein generation with regularized latent space optimization. *Nat. Mac. Intell.*, 4(10):840–851, 2022. doi: 10.1038/S42256-022-00532-1.

Chen, A., Dohan, D. M., and So, D. R. Evoprompting: Language models for code-level neural architecture search, 2023.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Chen, T., Chen, X., Chen, W., Heaton, H., Liu, J., Wang, Z., and Yin, W. Learning to optimize: A primer and a benchmark. *Journal of Machine Learning Research*, 23 (189):1–59, 2022a.

Chen, Y., Hoffman, M. W., Colmenarejo, S. G., Denil, M., Lillicrap, T. P., Botvinick, M., and Freitas, N. Learning to learn without gradient descent by gradient descent. In *International Conference on Machine Learning*, pp. 748–756. PMLR, 2017.

Chen, Y., Song, X., Lee, C., Wang, Z., Zhang, Q., Dohan, D., Kawakami, K., Kochanski, G., Doucet, A., Ranzato, M., Perel, S., and de Freitas, N. Towards learning universal

hyperparameter optimizers with transformers. In *Neural Information Processing Systems (NeurIPS) 2022*, 2022b.

Chen, Y., Song, X., Lee, C., Wang, Z., Zhang, R., Dohan, D., Kawakami, K., Kochanski, G., Doucet, A., Ranzato, M., et al. Towards learning universal hyperparameter optimizers with transformers. *Advances in Neural Information Processing Systems*, 35:32053–32068, 2022c.

Cheng, Z., Kasai, J., and Yu, T. Batch prompting: Efficient inference with large language model apis. *arXiv preprint arXiv:2301.08721*, 2023.

Droste, S., Jansen, T., and Wegener, I. Upper and lower bounds for randomized search heuristics in black-box optimization. *Theory of computing systems*, 39(4):525–544, 2006.

Eggensperger, K., Müller, P., Mallik, N., Feurer, M., Sass, R., Klein, A., Awad, N., Lindauer, M., and Hutter, F. Hpobench: A collection of reproducible multi-fidelity benchmark problems for hpo. *arXiv preprint arXiv:2109.06716*, 2021.

Elhara, O., Varelas, K., Nguyen, D., Tusar, T., Brockhoff, D., Hansen, N., and Auger, A. Coco: the large scale black-box optimization benchmarking (bbob-largescale) test suite. *arXiv preprint arXiv:1903.06396*, 2019.

Facebook. Adaptive experimentation platform, 2021. URL https://ax.dev/.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. Efficient and robust automated machine learning. *Advances in neural information processing systems*, 28, 2015.

Flood, M. M. The traveling-salesman problem. *Operations research*, 4(1):61–75, 1956.

Fu, C., Chen, P., Shen, Y., Qin, Y., Zhang, M., Lin, X., Yang, J., Zheng, X., Li, K., Sun, X., et al. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*, 2023.

Girdhar, R., El-Nouby, A., Liu, Z., Singh, M., Alwala, K. V., Joulin, A., and Misra, I. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15180–15190, 2023.

Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J., and Sculley, D. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pp. 1487–1495. ACM, 2017. doi: 10.1145/3097983.3098043.

Hansen, N. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

Hopcroft, J. E. and Ullman, J. D. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979. ISBN 0-201-02988-X.

Hutter, F., Hoos, H. H., and Leyton-Brown, K. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization: 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers 5*, pp. 507–523. Springer, 2011.

Ji, Z., Ma, P., Li, Z., and Wang, S. Benchmarking and explaining large language model-based code generation: A causality-centric approach. *arXiv preprint arXiv:2310.06680*, 2023.

Jin, H., Han, X., Yang, J., Jiang, Z., Liu, Z., Chang, C.-Y., Chen, H., and Hu, X. Llm maybe longlm: Self-extend llm context window without tuning. *arXiv preprint arXiv:2401.01325*, 2024.

Krishnamurthy, A., Harris, K., Foster, D. J., Zhang, C., and Slivkins, A. Can large language models explore in-context?, 2024.

Kudo, T. and Richardson, J. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

Lange, R., Schaul, T., Chen, Y., Lu, C., Zahavy, T., Dalibard, V., and Flennerhag, S. Discovering attention-based genetic algorithms via meta-black-box optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 929–937, 2023a.

Lange, R., Schaul, T., Chen, Y., Zahavy, T., Dalibard, V., Lu, C., Singh, S., and Flennerhag, S. Discovering evolution strategies via meta-black-box optimization. In *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, pp. 29–30, 2023b.

Lange, R. T., Tang, Y., and Tian, Y. Neuroevobench: Benchmarking evolutionary optimizers for deep learning applications. *arXiv preprint arXiv:2311.02394*, 2023c.

Lange, R. T., Tian, Y., and Tang, Y. Evolution transformer: In-context evolutionary optimization. *arXiv preprint arXiv:2403.02985*, 2024a.

Lange, R. T., Tian, Y., and Tang, Y. Large language models as evolution strategies. *arXiv preprint arXiv:2402.18381*, 2024b.

Lehman, J., Gordon, J., Jain, S., Ndousse, K., Yeh, C., and Stanley, K. O. Evolution through large models. In *Handbook of Evolutionary Machine Learning*, pp. 331–366. Springer, 2023.

Li, B., Mellou, K., Zhang, B., Pathuri, J., and Menache, I. Large language models for supply chain optimization. *arXiv preprint arXiv:2307.03875*, 2023a.

Li, D., Sun, Z., Hu, X., Liu, Z., Chen, Z., Hu, B., Wu, A., and Zhang, M. A survey of large language models attribution. *CoRR*, abs/2311.03731, 2023b. doi: 10.48550/ARXIV.2311.03731.

Li, K. and Malik, J. Learning to optimize. *arXiv preprint arXiv:1606.01885*, 2016.

Liu, F., Lin, X., Wang, Z., Yao, S., Tong, X., Yuan, M., and Zhang, Q. Large language model for multi-objective evolutionary optimization, 2023a.

Liu, F., Tong, X., Yuan, M., and Zhang, Q. Algorithm evolution using large language model, 2023b.

Liu, H., Li, C., Li, Y., and Lee, Y. J. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023c.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. In *NeurIPS*, 2023d.

Liu, S., Chen, C., Qu, X., Tang, K., and Ong, Y.-S. Large language models as evolutionary optimizers, 2023e.

Liu, T., Astorga, N., Seedat, N., and van der Schaar, M. Large language models to enhance bayesian optimization. *CoRR*, abs/2402.03921, 2024. doi: 10.48550/ARXIV.2402.03921.

Luo, H. and Specia, L. From understanding to utilization: A survey on explainability for large language models. *CoRR*, abs/2401.12874, 2024. doi: 10.48550/ARXIV.2401.12874.

Ma, Y. J., Liang, W., Wang, G., Huang, D.-A., Bastani, O., Jayaraman, D., Zhu, Y., Fan, L., and Anandkumar, A. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv:2310.12931*, 2023.

Madani, A., Krause, B., Greene, E. R., Subramanian, S., Mohr, B. P., Holton, J. M., Olmos, J. L., Xiong, C., Sun, Z. Z., Socher, R., Fraser, J. S., and Naik, N. Large language models generate functional protein sequences across diverse families. *Nat Biotechnol*, 41(8):1099–1106, Aug 2023.

Mashkaria, S. M., Krishnamoorthy, S., and Grover, A. Generative pretraining for black-box optimization. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 24173–24197. PMLR, 2023.

Mellor, J., Turner, J., Storkey, A., and Crowley, E. J. Neural architecture search without training. In *International Conference on Machine Learning*, pp. 7588–7598. PMLR, 2021.

Meyerson, E., Nelson, M. J., Bradley, H., Moradi, A., Hoover, A. K., and Lehman, J. Language model crossover: Variation through few-shot prompting, 2023.

Müller, S., Hollmann, N., Arango, S. P., Grabocka, J., and Hutter, F. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2021.

Müller, S., Feurer, M., Hollmann, N., and Hutter, F. Pfns4bo: In-context learning for bayesian optimization. *arXiv preprint arXiv:2305.17535*, 2023.

Nasir, M. U., Earle, S., Togelius, J., James, S., and Cleghorn, C. W. Llmatic: Neural architecture search via large language models and quality-diversity optimization. *CoRR*, abs/2306.01102, 2023. doi: 10.48550/ARXIV.2306.01102. URL https://doi.org/10.48550/arXiv.2306.01102.

Nguyen, T. and Grover, A. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S. (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16569–16594. PMLR, 2022.

Nguyen, T., Agrawal, S., and Grover, A. Expt: Synthetic pretraining for few-shot experimental design. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Nie, A., Cheng, C.-A., Kolobov, A., and Swaminathan, A. Importance of directional feedback for llm-based optimizers. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.

Oh, C., Tomczak, J. M., Gavves, E., and Welling, M. Combinatorial bayesian optimization using the graph cartesian product. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32:*

Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pp. 2910–2920, 2019.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi: 10.48550/ARXIV.2303.08774.

Perrone, V., Jenatton, R., Seeger, M. W., and Archambeau, C. Scalable hyperparameter transfer learning. *Advances in neural information processing systems*, 31, 2018.

Pham, A., Yang, C., Sheng, S., Zhao, S., Lee, S., Jiang, B., Dong, F., Guan, X., and Ming, F. OpenLLM: Operating LLMs in production, June 2023. URL https://github.com/bentoml/OpenLLM.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Rajendran, J., Irpan, A., and Jang, E. Meta-learning requires meta-augmentation. *Advances in Neural Information Processing Systems*, 33:5705–5715, 2020.

Ramamurthy, R., Ammanabrolu, P., Brantley, K., Hessel, J., Sifa, R., Bauckhage, C., Hajishirzi, H., and Choi, Y. Is reinforcement learning (not) for natural language processing?: Benchmarks, baselines, and building blocks for natural language policy optimization. *arXiv preprint arXiv:2210.01241*, 2022.

Real, E., Liang, C., So, D., and Le, Q. Automl-zero: Evolving machine learning algorithms from scratch. In *International conference on machine learning*, pp. 8007–8019. PMLR, 2020.

Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., Ruiz, F. J., Ellenberg, J. S., Wang, P., Fawzi, O., et al. Mathematical discoveries from program search with large language models. *Nature*, pp. 1–3, 2023.

Ru, B. X., Wan, X., Dong, X., and Osborne, M. A. Interpretable neural architecture search via bayesian optimisation with weisfeiler-lehman kernels. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Sha, H., Mu, Y., Jiang, Y., Chen, L., Xu, C., Luo, P., Li, S. E., Tomizuka, M., Zhan, W., and Ding, M. Languagempc: Large language models as decision makers for autonomous driving. *arXiv preprint arXiv:2310.03026*, 2023.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and de Freitas, N. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104 (1):148–175, 2016. doi: 10.1109/JPROC.2015.2494218.

Shen, S., Hou, L., Zhou, Y., Du, N., Longpre, S., Wei, J., Chung, H. W., Zoph, B., Fedus, W., Chen, X., et al. Mixture-of-experts meets instruction tuning: A winning combination for large language models. *arXiv preprint arXiv:2305.14705*, 2023.

Snoek, J., Larochelle, H., and Adams, R. P. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

Song, X., Li, O., Lee, C., Yang, B., Peng, D., Perel, S., and Chen, Y. Omnipred: Language models as universal regressors. *CoRR*, abs/2402.14547, 2024. doi: 10.48550/ARXIV.2402.14547.

Swersky, K., Snoek, J., and Adams, R. P. Multi-task bayesian optimization. *Advances in neural information processing systems*, 26, 2013.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Canton-Ferrer, C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023. doi: 10.48550/ARXIV.2307.09288.

Trabucco, B., Geng, X., Kumar, A., and Levine, S. Design-bench: Benchmarks for data-driven offline model-based optimization. In *International Conference on Machine Learning*, pp. 21658–21676. PMLR, 2022.

Turner, R., Eriksson, D., McCourt, M., Kiili, J., Laaksonen, E., Xu, Z., and Guyon, I. Bayesian optimization is

superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. In *NeurIPS 2020 Competition and Demonstration Track*, pp. 3–26. PMLR, 2021.

Unterthiner, T., Keysers, D., Gelly, S., Bousquet, O., and Tolstikhin, I. O. Predicting neural network accuracy from weights. *CoRR*, abs/2002.11448, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017.

Volpp, M., Fröhlich, L. P., Fischer, K., Doerr, A., Falkner, S., Hutter, F., and Daniel, C. Meta-learning acquisition functions for transfer learning in bayesian optimization. *arXiv preprint arXiv:1904.02642*, 2019.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Wang, J., Liu, Z., Zhao, L., Wu, Z., Ma, C., Yu, S., Dai, H., Yang, Q., Liu, Y., Zhang, S., Shi, E., Pan, Y., Zhang, T., Zhu, D., Li, X., Jiang, X., Ge, B., Yuan, Y., Shen, D., Liu, T., and Zhang, S. Review of large vision models and visual prompt engineering. *Meta-Radiology*, 1(3): 100047, 2023a. ISSN 2950-1628. doi: https://doi.org/10.1016/j.metrad.2023.100047.

Wang, J., Shi, E., Yu, S., Wu, Z., Ma, C., Dai, H., Yang, Q., Kang, Y., Wu, J., Hu, H., et al. Prompt engineering for healthcare: Methodologies and applications. *arXiv preprint arXiv:2304.14670*, 2023b.

Wang, Z., Dahl, G. E., Swersky, K., Lee, C., Mariet, Z., Nado, Z., Gilmer, J., Snoek, J., and Ghahramani, Z. Pre-trained gaussian processes for bayesian optimization. *arXiv preprint arXiv:2109.08215*, 2021.

Wistuba, M. and Grabocka, J. Few-shot bayesian optimization with deep kernel surrogates. *arXiv preprint arXiv:2101.07667*, 2021.

Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. *arXiv*, 2023.

Xiong, M., Hu, Z., Lu, X., Li, Y., Fu, J., He, J., and Hooi, B. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. *CoRR*, abs/2306.13063, 2023. doi: 10.48550/ARXIV.2306.13063.

Xu, F. F., Alon, U., Neubig, G., and Hellendoorn, V. J. A systematic evaluation of large language models of code. In *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, pp. 1–10, 2022.

Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. Large language models as optimizers, 2023.

Ying, C., Klein, A., Christiansen, E., Real, E., Murphy, K., and Hutter, F. Nas-bench-101: Towards reproducible neural architecture search. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7105–7114. PMLR, 2019.

Yogatama, D. and Mann, G. Efficient transfer learning method for automatic hyperparameter tuning. In *Artificial intelligence and statistics*, pp. 1077–1085. PMLR, 2014.

Yu, W., Gileadi, N., Fu, C., Kirmani, S., Lee, K.-H., Arenas, M. G., Chiang, H.-T. L., Erez, T., Hasenclever, L., Humplik, J., et al. Language to rewards for robotic skill synthesis. *arXiv preprint arXiv:2306.08647*, 2023.

Zhang, M., Desai, N., Bae, J., Lorraine, J., and Ba, J. Using large language models for hyperparameter optimization. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.