Research
A SCIENCE PARTNER JOURNAL

REVIEW ARTICLE

# When Large Language Models Meet Evolutionary Algorithms: Potential Enhancements and Challenges

**Chao Wang, Jiaxuan Zhao, Licheng Jiao*, Lingling Li, Fang Liu, and Shuyuan Yang**

School of Artificial Intelligence, Xidian University, Xi'an 710071, Shaanxi, China.

*Address correspondence to: lchjiao@mail.xidian.edu.cn

**Citation:** Wang C, Zhao J, Jiao L, Li L, Liu F, Yang S. When Large Language Models Meet Evolutionary Algorithms: Potential Enhancements and Challenges. *Research* 2025;8:Article 0646. https://doi.org/10.34133/research.0646

Submitted 10 November 2024
Revised 3 March 2025
Accepted 4 March 2025
Published 27 March 2025

Pre-trained large language models (LLMs) exhibit powerful capabilities for generating natural text. Evolutionary algorithms (EAs) can discover diverse solutions to complex real-world problems. Motivated by the common collective and directionality of text generation and evolution, this paper first illustrates the conceptual parallels between LLMs and EAs at a micro level, which includes multiple one-to-one key characteristics: token representation and individual representation, position encoding and fitness shaping, position embedding and selection, Transformers block and reproduction, and model training and parameter adaptation. These parallels highlight potential opportunities for technical advancements in both LLMs and EAs. Subsequently, we analyze existing interdisciplinary research from a macro perspective to uncover critical challenges, with a particular focus on evolutionary fine-tuning and LLM-enhanced EAs. These analyses not only provide insights into the evolutionary mechanisms behind LLMs but also offer potential directions for enhancing the capabilities of artificial agents.

## Introduction

Natural language processing (NLP) focuses on enabling computers to understand, generate, and process human language, covering tasks such as text generation [1], text segmentation [2], named entity recognition [3], sequence labeling [4], and relation extraction [5]. Large language models (LLMs), such as generative pre-trained Transformer (GPT) [6] and bidirectional encoder representations from transformer (BERT) [7], primarily learn statistical patterns with temporal relations from text sequences, often in an unsupervised manner, to establish probability distributions of texts. These models have become foundational tools for the aforementioned NLP tasks. By analyzing input tokens and iteratively generating the most likely subsequent tokens, LLMs like GPT and BERT can produce coherent and contextually relevant texts. This sequence-to-sequence model with powerful understanding and generation capabilities has been employed to assist users on a variety of innovative tasks, including writing, mathematical discovery, and chemical research [8–13]. Meanwhile, training LLMs on vast text demands substantial computing resources, as exemplified by ChatGPT's pre-training consumption of several thousand petaflop/s-days [8]. Fine-tuning techniques have been proposed to alleviate the computational challenges of training from scratch [14]. In a model-as-a-service scenario [15], LLMs are only accessed as application programming interfaces (APIs) for inference. Because of their gradient-free nature, evolutionary algorithms (EAs) are employed to fine-tune LLMs in black-box scenarios, where they rely solely on forward propagation and do not require access to internal model gradients [15]. This makes EAs a practical choice for such settings.

Drawing from biological evolution, EAs continuously maintain evolving systems (population or probability distribution) through reproduction and selection to explore fitness landscapes [16–18]. Typical methods include the genetic algorithm (GA), evolutionary strategy (ES), and genetic programming (GP) [19,20]. In principle, only individuals and their fitness are needed to drive the evolutionary process in these approaches. Due to advancements in computational resources, EAs have provided diverse solutions to complex black-box optimization issues, such as neuroevolution [21], industrial design [22–25], and natural sciences [26]. Nonetheless, most EAs are task-specific, and their capabilities do not automatically increase with experience [27–30]. Recently, Transformer-enhanced EAs utilize basic Transformer models [31] to learn optimization experiences, while LLM-enhanced EAs employ well-trained LLMs [32,33] to produce optimization experiences. Both approaches intend to improve the performance and generalization of EAs.

Figure 1 demonstrates that text and population can be regarded as sequence data, specifically exhibiting directionality. In a text, each token corresponds to a specific position, while in a population, each individual is associated with a particular fitness rank. Text sequences have a natural directionality derived from human-defined grammatical rules. Population evolution is also directional, primarily driven by fitness-based selection. LLMs and EAs are designed to learn or simulate such sequential data (text sequence and population sequence). Figure 2 illustrates the process of the sequences generated by LLMs and EAs, taking the GPT [6] and GA [34] as examples, respectively. GPT continuously generates subsequent tokens by iterating over a context window. The tokens in the input window are transformed by a

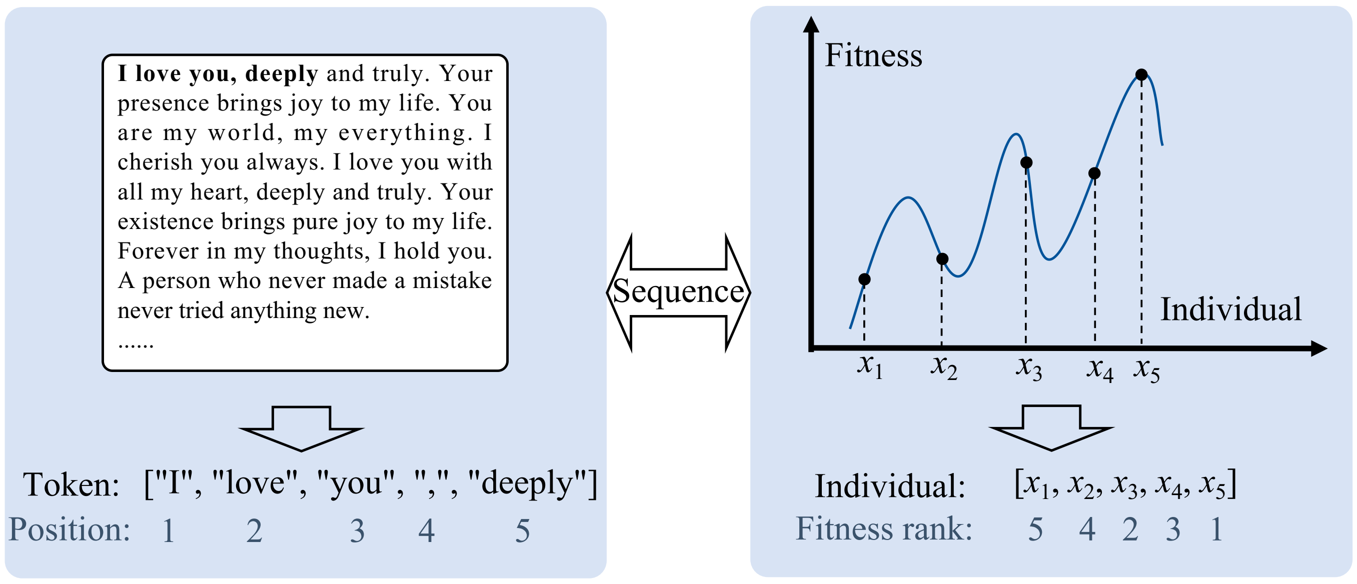

I love you, deeply and truly. Your presence brings joy to my life. You are my world, my everything. I cherish you always. I love you with all my heart, deeply and truly. Your existence brings pure joy to my life. Forever in my thoughts, I hold you. A person who never made a mistake never tried anything new. ......
Token: ["I", "love", "you", ",", "deeply"]
Position: 1 2 3 4 5
Sequence
Fitness
Individual
$x_1$ $x_2$ $x_3$ $x_4$ $x_5$
Individual: [$x_1$, $x_2$, $x_3$, $x_4$, $x_5$]
Fitness rank: 5 4 2 3 1

**Fig. 1.** Both tokens in a text and individuals in a population can be regarded as sequences.

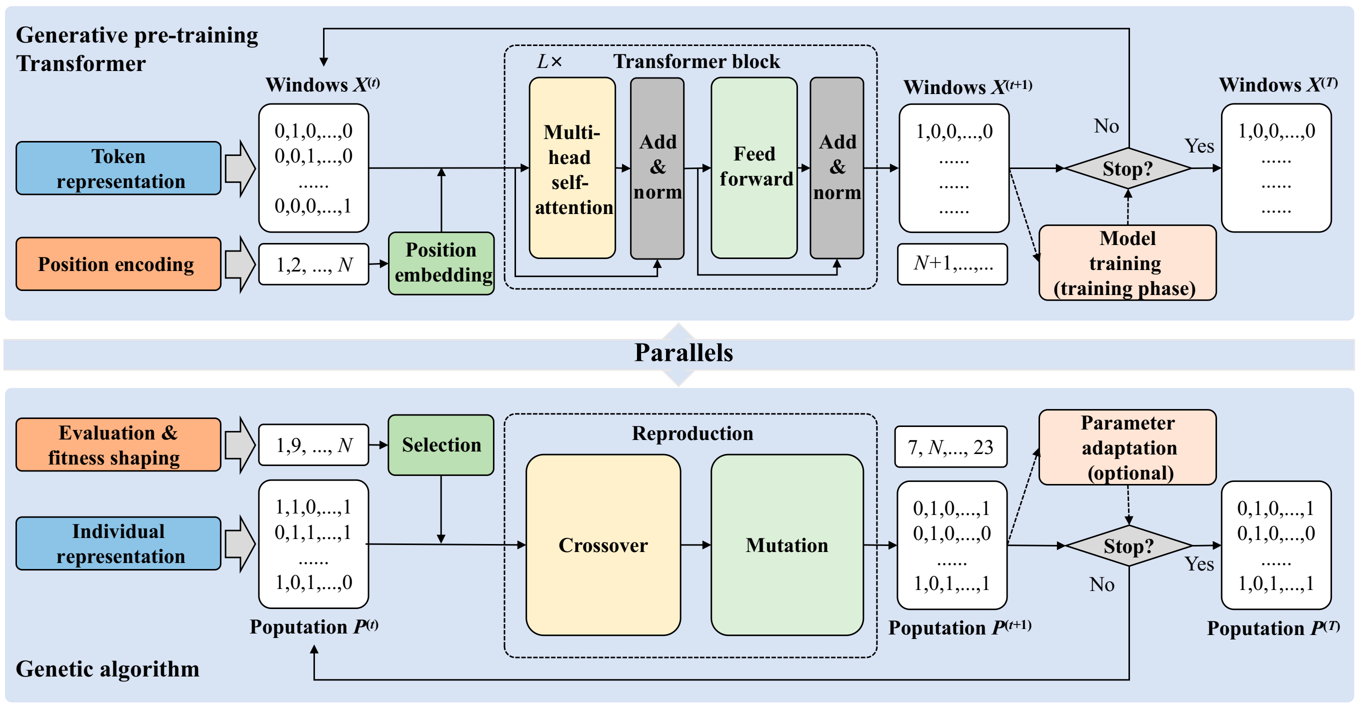


**Fig. 2.** Overview of the generative pre-training Transformer (GPT) and genetic algorithm (GA). Modules of the same color indicate parallels, as exemplified by the analogy between crossover in GA and attention in GPT.

large-scale Transformer block [multi-head self-attention and feed-forward neural network (FFN)]. These tokens function collectively, providing contextual information for accurate generation. GA generates individuals with high fitness by iterating a population. Individuals in the parent population are transformed by reproduction operators (crossover and mutation). These individuals exhibit collective intelligence, helping to explore the search space and seek the optimal solutions. During the generation process, both the context window in GPT and the population in GA are continuously updated to produce coherent texts and diverse solutions, respectively. Notably, the context window and population share a collective nature, where their constituent elements (tokens in GPT and individuals in GA) function cohesively within their respective domains.

Inspired by the above directionality and collective, we raise the following issues: Are there parallels between LLMs and EAs? How can these parallels be utilized to address existing limitations and foster innovation in the coupling of LLMs and EAs? Moreover, since current interdisciplinary research focusing on either evolutionary fine-tuning or LLM-enhanced EAs remains in its infancy, what are the key challenges faced by these efforts? To address these issues, this paper draws conceptual analogies between the primary characteristics of LLMs and EAs, emphasizing their common mechanisms. At the micro level, we analyze key interdisciplinary research related to each parallel, which not only supports our analogies but also offers insights for potential improvements. At the macro level, we systematically summarize evolutionary fine-tuning and LLM-enhanced EAs to reveal critical challenges. The main contributions of this paper are as follows:

1. At the micro level, we analyze existing research by drawing conceptual analogies between the key characteristics of LLMs and EAs, aiming to inspire novel ideas and technologies that can advance these fields.
2. At the macro level, we provide the first systematic overview of evolutionary fine-tuning and LLM-enhanced EAs, highlighting key challenges for future research.

The remainder of this paper is organized as follows. The "Parallels" section presents the conceptual parallels between LLMs and EAs from 5 perspectives, introducing potential technical improvements. In the "Evolutionary Fine-Tuning in Black-Box Scenarios" and "LLM-Enhanced EA" sections, we summarize evolutionary fine-tuning in black-box scenarios and LLM-enhanced EAs, respectively. Additionally, key future challenges are identified. Finally, the "Conclusion" section summarizes the work in this paper.

## Parallels

Transformer-based LLMs have developed rapidly since the introduction of the Transformer architecture [35]. Taking GPT as an example, LLMs mainly contain several characteristics: token representation, position encoding, position embedding ("position encoding" and "position embedding" are closely related in some literature; for the sake of analogy, we have chosen to treat them separately), Transformer block, and model training. In 1950, Turing proposed a "learning machine" similar to the principles of evolution [36]. Since then, evolution-inspired computational theories have been explored and refined. Taking GA as an example, EAs include several typical characteristics: individual representation, fitness shaping, selection, reproduction, and parameter adaptation. Table 1 lists these characteristics, classic methods, traits, and key interdisciplinary research integrating LLMs and EAs. Next, each subsection focuses on elucidating the corresponding characteristics of LLMs and EAs.

### Token representation and individual representation

In LLMs, the input is represented as a token sequence $X = \left\{x_1, \ldots, x_N\right\} \in \mathbb{R}^{N \times |V|}$, where $N$ and $|V|$ are the sizes of context window and vocabulary $V$, respectively. Each token is encoded as a high-dimensional sparse one-hot vector. Subsequently, token embeddings map token encoding sequences into a low-dimensional dense vector space [6]. For example, linear transformation applies a word embedding matrix $W_e \in \mathbb{R}^{|V| \times d_t}$ to the token encoding sequence $X$ to generate a new representation $X = XW_e$.

In EAs, the population is represented as an individual sequence $P = \left\{p_i, \ldots, p_N\right\} \in \mathbb{R}^{N \times d}$, where $N$ and $d$ are the population size and coding dimension, respectively. Each individual (or solution) is encoded into a data structure manipulable by genetic operators. In numerical optimization, real encoding maps the individual into a real-valued vector. The high dimensionality of encoding increases optimization difficulty. Many strategies are proposed to deal with the curse of dimensionality [37–39]. For example, random embedding applies a random projection matrix $W_r \in \mathbb{R}^{|V| \times d_r}$ to the population $P$ to generate a low-dimensional representation $P = PW_r$.

Token representation can be regarded as an individual representation, which satisfies collective and uniqueness. The tokens in the context window are individuals in the population. Both token encoding and individual encoding guarantee one-to-one mapping.

This analogy conceptually provides bidirectional support for interdisciplinary research. EAs using token representations can operate directly within embedded or original token spaces to find high-quality input prompts [15,40]. Natural language on a finite vocabulary has demonstrated powerful representation capabilities, which may bring new opportunities for individual representation. In evolution, the decision space may be infinite, changing, and difficult to describe mathematically. Fortunately, these complex search spaces are represented directly with the help of natural language. This flexibility enables LLM-enhanced EAs to tackle tasks that are not easily reducible to simple mathematical formulas or notations, such as paths and coding [33,41].

### Position encoding and fitness shaping

Position encoding models the dependence of tokens at different positions in the sequence. In GPT [6], sine and cosine functions of different frequencies are adopted to encode dependencies. Each token has a unique position encoding consisting of cosine functions with different frequencies. The combination of several cosine waves contains relative distance information between tokens. However, due to the symmetry of distances, the position encoding cannot distinguish sequence direction. Notably, Lyu et al. [42] investigate whether models can learn directionality, emphasizing its importance for interpretability and performance.

Fitness shaping transforms the fitness of individuals in a population to cope with selection pressure. For example, rank-based fitness shaping is commonly used in ES [43,44]. Individuals are sorted in descending order of fitness. The corresponding fitness is transformed into a set of utility values $u_1 \geq u_2 \geq \ \ldots \ \geq u_N$ by a utility function. This utility function ensures invariance under fitness order-preserving transformations, which preserves the relativity and directionality of fitness.

Both position encoding and fitness shaping share the characteristics of coding uniqueness. The position encoding used in GPT effectively models the relative positions between tokens, although it does not explicitly capture the directionality of the sequence. Fitness shaping describes the relative and directional ordering of individual fitness. Inspired by fitness shaping, the integration of sequence directionality into position encoding emerges as a noteworthy research direction. Drawing from fitness shaping techniques in CMA-ES/NES [43,44], future work could design position encodings that preserve order invariance, capturing both the relative positions and directionality of tokens in text. Such methods could improve performance in tasks like long-text generation and question answering by ensuring that the model respects the temporal and contextual order of information. Conversely, in existing Transformer-enhanced EAs [31], fitness values are directly utilized for position encoding

**Table 1.** A comparison of large language models and evolutionary algorithms in terms of key characteristics, where "N/A" indicates a lack of corresponding interdisciplinary research in that area

| Large language models | | | | | Evolutionary algorithms | | | |
|---|---|---|---|---|---|---|---|---|
| Section | Characteristics | Classic methods | Traits | Key inter-disciplinary research | Characteristics | Classic methods | Traits | Key interdisciplinary research |
| Token representation and individual representation | Token representation | One-hot encoding + linear transformation | Collective, uniqueness, finite | [15,40] | Individual representation | Real encoding + random embedding [37] | Collective, uniqueness, infinite and changing | [33,41] |
| Position encoding and fitness shaping | Position encoding | Sine and cosine functions [6] | Uniqueness, relativity | N/A | Fitness shaping | Rank transformation, utility function [43] | Uniqueness, relativity, directionality | [31] |
| Position embedding and selection | Position embedding | Absolute, relative, rotary [56] | Relativity | N/A | Selection | Tournament selection, rank selection | Relativity, directionality | [31] |
| Transformer block and reproduction | Transformer block | Multi-head self-attention + feed-forward neural network | Position-insensitive, parallelism, sparsity, token and position information, singleness, synergy | [51] | Reproduction | Arithmetic crossover + uniform mutation | Fitness-insensitive, parallelism, sparsity, individual and fitness information, singleness, synergy | [31,52] |
| Model training and parameter adaptation | Model training | Pre-training, fine-tuning, reinforcement learning | Learning, exploration, parameter space, language space | [15,40] | Parameter adaptation | Hyper-heuristics, pre-training, fine-tuning, meta-learning | Learning, exploration, parameter space, search space | [31,33,41,69,70] |

within the Transformer model. According to our conceptual analogy, the introduction of fitness shaping has the potential to aid Transformer-enhanced EAs in managing selective pressures.

## Transformer block and reproduction

A vanilla Transformer block is composed of a multi-head self-attention attention, an FFN, residual connections, and layer normalization. The self-attention attention mechanism performs feature transformation on the token embedding. Token embedding $X \in \mathbb{R}^{N \times d_t}$ is transformed into query $Q = XW^Q$, key $K = XW^K$, and value $V = XW^V$ through linear transformations $W^Q \in \mathbb{R}^{d_t \times d_q}$, $W^K \in \mathbb{R}^{d_t \times d_k}$, and $W^V \in \mathbb{R}^{d_t \times d_v}$. Then, query $Q$ and key $K$ are used to calculate the attention matrix $A$ describing token relationships. Applying the Softmax function to $A$ and multiplying by value $V$ yields the transformed output $X'$:

$$A' = QK^T, A = \text{Softmax}\left(\frac{A'}{\sqrt{d}}\right), X' = AV \tag{1}$$

The multi-head self-attention mechanism operates by combining multiple self-attention mechanisms to focus on information from different subspaces. Many studies show that the learned attention matrix is sparse [45–47]. To address the computational cost, various improvement mechanisms have been proposed, such as sparse attention and linear attention [48]. In Transformer blocks, the FFN enhances the expressive ability of each token $x_i'$ by applying a nonlinear function $f\left(x_i'\right)$. Residual connections help alleviate the vanishing gradient problem, enabling deeper feature learning. Layer normalization stabilizes the training process and improves convergence speed.

Typical reproduction involves crossover and mutation. Crossover acts on the parent population to generate new individuals. Classic real crossover operators include arithmetic crossover, simulated binary crossover [49], and more. We illustrate crossover's workflow using arithmetic crossover [50] as an example. Any 2 individuals $p_i$ and $p_j$ are randomly selected from the parent population $P \in \mathbb{R}^{N \times d}$. New individual $p_i'$ is a linear combination of parental genes:

$$p_i' = a_i p_i + a_j p_j \tag{2}$$

We reformulate this process as:

$$\begin{aligned} p_i' &= 0p_1 + \ldots + a_i p_i + \ldots + a_j p_j + \\ 0p_N &= \left[0, \ldots, a_i, \ldots, a_j, \ldots\right] P = A_i P \end{aligned} \tag{3}$$

Without loss of generality, $N$ individuals can be produced in a batch:

$$P' = \left[A_1; A_2; \ldots; A_N\right] P = AP \tag{4}$$

where $A$ is a sparse matrix determined by the selection, termed the selection matrix in the paper. In the reproduction, mutation applies a nonlinear perturbation $PM\left(p_i'\right)$ to each individual $p_i'$ to promote individual diversity.

Comparing Eq. 1 with Eq. 4, attention and crossover share a similar mathematical representation, exhibiting parallelism and sparsity. Attention does not explicitly model token positions. Similarly, crossover inherently does not consider individual fitness. The attention and selection matrices play analogous roles: One determines token feature combinations, while the other governs parent genetic combinations. The attention matrix is parameterized based on token embeddings, while the selection matrix is heuristically built on individual relationships. Additionally, Eq. 1 demonstrates that the input and output of attention occupy distinct latent spaces. However, Eq. 4 reveals that the population keeps the same representation space across crossover. This analogy highlights opportunities to borrow ideas from attention mechanisms to improve crossover operators and vice versa, as demonstrated in recent studies [51,52].

Both the FFN and mutation operate independently on each singleton (token or individual) and can be processed in parallel. Existing works [51,53] show that the removal of FFNs degrades Transformer performance, emphasizing the crucial role of both attention and FFNs. Analogously, the synergistic effect of crossover and mutation results in the super efficiency of EAs [54,55]. While FFNs and mutation are not mathematically equivalent, their functional roles in maintaining performance are conceptually similar. This analogy suggests potential opportunities to enhance FFNs by introducing controlled randomness, inspired by the role of mutation in generating diversity within populations in EAs.

Zhang et al. [51] first analogized Transformer blocks to reproduction, utilizing dynamic local populations in EAs to enhance Vision Transformers. Similarly, Li et al. [52] modeled crossover and mutation using attention and FFN mechanisms, respectively, to develop Transformer-enhanced EAs. These efforts highlight the functional parallels between Transformer components and EA operations, confirming our conceptual analogy and demonstrating the potential for progress through idea sharing between advanced Transformers and reproduction.

## Position embedding and selection

Position embeddings integrate positional information into the attention mechanism, using absolute, relative, and rotary techniques, to capture sequential dependencies and contextual relationships within token sequences [56]. A typical absolute position embedding is the sinusoidal position embedding, adding position information encoded by sine and cosine functions to token embeddings. For any 2 tokens $x_t$ and $x_s$, with position information $p_t$ and $p_s$, the attention matrix is expressed as:

$$A_{t,s} = Q_t^T K_s = \left(x_t + p_t\right)^T W_Q^T W_K \left(x_s + p_s\right) \tag{5}$$

$$= x_t^T W_Q^T W_K x_s + x_t^T W_Q^T W_K p_s + p_t^T W_Q^T W_K x_s + p_t^T W_Q^T W_K p_s \tag{6}$$

Due to $W_Q^T W_K$, the token relative information is destroyed [57]. Several models like T5 [58], Transformer-XL [59], TENER [57], and DeBERTa [60] have integrated relative positional information into the attention matrix to address this limitation. For example, T5 directly adds token offsets to attention weights:

$$A_{t,s} = Q_t^T K_s + r_{b(t-s)} = x_t^T W_Q^T W_K x_s + r_{b(t-s)} \tag{7}$$

Rotary position embedding incorporates relative position information through token embedding rotation [56]:

$$A_{t,s} = x_t W_Q R_{t-s} W_K^T x_s^T = x_t W_Q R_t \left( R_s^T W_K^T x_s^T \right) \tag{8}$$

which achieves a unification of absolute and relative position embeddings.

Based on fitness information, selection operators such as tournament and rank selection are designed to identify excellent parents for crossover [50]. These selection methods can be viewed as heuristics for building the selection matrix. In binary tournament selection, for instance, the selection matrix $A$ is randomly created based on fitness comparisons. In basic differential evolution [61], multiple individuals are randomly chosen for differential operations, which influences the composition of the selection matrix $A$. In OpenAI-ES [62], each individual is sampled from a multivariate Gaussian distribution with mean $\frac{1}{N}\sum_1^N u_i p_i$ and covariance $\sigma^2 I$:

$$p_i' = \frac{1}{N}\sum_1^N u_i p_i + N\left(0, \sigma^2 I\right) \tag{9}$$

where $u_i$ is the utility function of $i$th parent individual $p_i$. The first term is crossover, and the second is mutation. The crossover is rewritten as:

$$\frac{1}{N}\sum_1^N u_i p_i = \left[\frac{1}{N}u_1, \dots, \frac{1}{N}u_N\right] P = A_i P \tag{10}$$

where each row of the selection matrix $A_i$ is determined by the parent's utility value. In OpenAI-ES, the selection matrix has identical rows due to the same genetic material from parents assigned to each individual. Furthermore, in NSGAII [63], the selection matrix A is constructed using nondominated sorting and crowding distance, considering individual relationships in the objective space.

By operating on matrix $A$, position embedding and selection add position and fitness information to the attention mechanism and crossover, respectively. The selection notably steers the population toward enhanced fitness. Individuals with higher fitness are preferentially selected for crossover, which considers the directionality of individual fitness, fostering a more adaptive population. Standard position embedding effectively captures the relative positions between tokens but does not explicitly encode the sequential order of tokens. Current efforts introduced task-specific supervision during training to assist LLMs in comprehending the sequential relationships among tokens. For example, GPT [6] employs a masked multi-head attention mechanism, ensuring that the output at each position is solely determined by preceding tokens. This approach guarantees a unidirectional information flow, forcing the self-attention mechanism to consider only the past context. Masked LLMs like BERT [64] learn contextual representations by predicting masked tokens, necessitating a focus on the entire textual context in both directions rather than just a unilateral one. Inspired by the directionality of fitness considered in selection, introducing token order directly into position embedding may enhance the generative capabilities of LLMs.

The attention matrix is influenced by both token and position relations. The selection matrix is usually designed based on fitness relations. In complex fitness landscapes, additional considerations such as genetic similarity among individuals are factored into the selection matrix. For instance, in multi-modal optimization with multiple global optima [65,66], individual distances within the search space are employed as a criterion to preserve population diversity during selection. This ensures that the population does not converge prematurely to a single optimum. The selection, in this context, incorporates both individual relations (e.g., distances between individuals) and fitness relations (e.g., fitness rank). Essentially, the similarity between attention and selection matrices in handling relational information stems from the analogy drawn to tokens and individuals, as well as to positions and fitness. Existing Transformer-enhanced EA merges individual embeddings with fitness embeddings, echoing how token embeddings are combined with positional embeddings [31]. This practice serves as a support for our conceptual analogy. Advanced attention mechanisms, such as those incorporating rotational positional embedding [56], have the potential to enhance the performance of this operation in Transformer-enhanced EAs.

## Model training and parameter adaptation

Model training typically begins with unsupervised pre-training, modeling natural language on a vast amount of text. This is followed by fine-tuning, which adjusts the model to downstream tasks. Given a token sequence $X = \{x_1, \dots, x_N\}$, a unidirectional LLM estimates a conditional probability distribution $P\left(x_i | x_1, \dots, x_{i-1}; \theta\right)$ to generate subsequent tokens. For example, In GPT's pre-training [6], the goal of language modeling is to maximize the log likelihood:

$$L^{PT} = \sum_{i=1}^N \log P\left(x_i | x_1, \dots, x_N; \theta\right) \tag{11}$$

where $N$ is the context window size and $\theta$ is the model parameter. During fine-tuning, the optimization objective is a weighted sum of pre-training loss $L^{PT}$ and fine-tuning loss $L^{FT}$ [6]:

$$L = L^{FT} + \mu L^{PT} \tag{12}$$

Hyperparameter $\mu \in [0,1]$ determines the trade-off across losses. Furthermore, reinforcement learning [67] fine-tunes LLMs by optimizing outputs' overall performance (rewards) to generate high-quality responses continuously. Evolutionary fine-tuning is proposed for black-box cases with inaccessible gradients and limited resources. These methods typically guide LLMs to generate the desired output by automatically constructing prompts directly within the input sequence [15,40].

In GA, given parent population $P = \{p_1, \dots, p_N\}$, offspring are sampled from an implicit conditional probability distribution $P\left(p_i | p_1, \dots, p_N\right)$, which is induced by genetic operators including selection, crossover, and mutation [41]. Genetic operator parameters are often determined through repeated experiments or adaptively updated using hyper-heuristic strategies [68]. In ES, the probability distribution $P\left(p | p_1, \dots p_N; \theta\right)$ over the parent population is employed to produce offspring. For example, in CMAES [44], the parameters (mean and covariance matrix) of a multivariate Gaussian distribution are adapted by maximizing the log likelihood:

$$\sum_{i=1}^N u_i \log P\left(p_{i:N}; m\right);\ \sum_{i=1}^N u_i \log P\left(\frac{p_{i:N} - m}{\sigma}; C\right) \tag{13}$$

where $p_{i:N}$ refers to the $i$th ranked individual in a population with $N$ individuals based on fitness. The first term is the mean update, while the second term is the ranking-$N$ update of the covariance matrix. Recently, Transformer-enhanced EAs adaptively updated parameters from optimization experiences on a set of optimization tasks, improving the generalization ability of EAs on new tasks. Common methods include pre-training [31,33] and meta-learning [69,70]. In pre-training, optimization experiences for multi-objective optimization [31] consist of the population and their fitness generated by multi-objective EAs on numerous benchmarks. Optimization experiences for GP [33] include incremental changes in files submitted by humans to version control systems like Github. Meta-learning [69,70] adaptively updates parameters by optimizing the average performance of EAs across a set of tasks. Regrettably, no comprehensive study has compared these 2 methods within a single framework. Furthermore, well-trained LLMs are directly utilized as reproduction operators with human-like experience [33,71]. Prompts based on historical populations are constructed to guide LLMs in generating the desired output population [33,41].

Despite differing implementation strategies, LLMs and EAs converge on a shared fundamental objective: revealing the underlying probability distributions within data, thereby facilitating the learning and exploration of knowledge. In pre-training and supervised fine-tuning, LLMs construct conditional probability distributions through the accurate prediction of tokens, learning vast pre-existing knowledge. Reinforcement learning adjusts the LLM parameters based on the rewards. Evolutionary fine-tuning automatically searches for high-quality prompts or configurations to improve output quality [15,72]. These 2 paradigms explore new knowledge specific to the target task. Learning and exploration jointly ensure the generative and generalization abilities of LLMs. EAs shape probability distributions based on fitness evaluated via real-time individual–environment interaction. In Transformer/LLM-enhanced EAs, models learn from existing optimization experiences or human-like experiences, endowing EAs with powerful learning capabilities. Additionally, LLM training works in the parameter space, while evolutionary fine-tuning extends to the language space. EAs operate in both the search space (e.g., GA) and the parameter space (e.g., CMAES). The aforementioned analogy provides a reasonable motivation for interdisciplinary research: Merging the exceptional learning capabilities of LLMs with the remarkable exploration abilities of EAs can foster advancements in their respective fields.

In Eq. 12, the hyperparameter $\mu$ is carefully tuned manually, as it affects the generalization ability of LLMs. The loss trade-off is modeled as a multi-objective optimization problem [73,74]. Applying advanced multi-objective EAs aids in creating stronger supervised fine-tuning paradigms. Recent studies [62] show that ES has advantages over gradient-based reinforcement learning for long episodes with very many time steps. ES is promising as an alternative to reinforcement learning for training LLMs in multi-turn dialogue systems.

The generalization of the Transformer-enhanced EAs is influenced by optimization experience, which involves a set of historical optimization tasks. The similarity between historical and new tasks determines the effectiveness of optimization experience utilization, echoing the motivation behind evolutionary transfer optimization (ETO) [27]. Benchmarks for ETO [28] can potentially serve as optimization experiences for training in diverse transfer scenarios. Additionally, using algorithms, human expertise, or LLMs to generate optimization experiences across various benchmarks is critical for expanding the application scope.

## Summary

Despite their independent development, LLMs and EAs share certain conceptual parallels. The parallels have inspired novel ideas and technical advancements, as outlined in Fig. 3. From a macro perspective, the parallels between LLMs and EAs provide a conceptual framework that can inspire the development of artificial agents capable of learning from established knowledge while

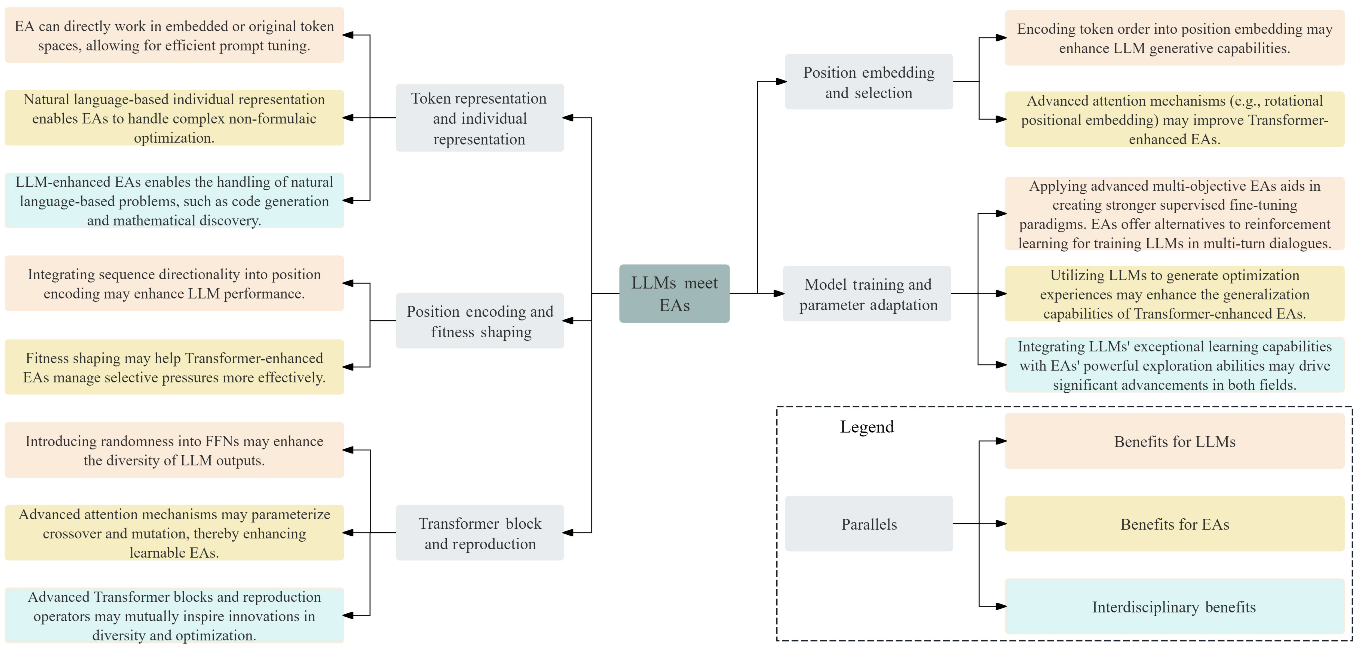

EA can directly work in embedded or original token spaces, allowing for efficient prompt tuning.
Natural language-based individual representation enables EAs to handle complex non-formulaic optimization.
LLM-enhanced EAs enables the handling of natural language-based problems, such as code generation and mathematical discovery.
Integrating sequence directionality into position encoding may enhance LLM performance.
Fitness shaping may help Transformer-enhanced EAs manage selective pressures more effectively.
Introducing randomness into FFNs may enhance the diversity of LLM outputs.
Advanced attention mechanisms may parameterize crossover and mutation, thereby enhancing learnable EAs.
Advanced Transformer blocks and reproduction operators may mutually inspire innovations in diversity and optimization.
Token representation and individual representation
Position encoding and fitness shaping
Transformer block and reproduction
LLMs meet EAs
Position embedding and selection
Model training and parameter adaptation
Encoding token order into position embedding may enhance LLM generative capabilities.
Advanced attention mechanisms (e.g., rotational positional embedding) may improve Transformer-enhanced EAs.
Applying advanced multi-objective EAs aids in creating stronger supervised fine-tuning paradigms. EAs offer alternatives to reinforcement learning for training LLMs in multi-turn dialogues.
Utilizing LLMs to generate optimization experiences may enhance the generalization capabilities of Transformer-enhanced EAs.
Integrating LLMs' exceptional learning capabilities with EAs' powerful exploration abilities may drive significant advancements in both fields.
Legend
Parallels
Benefits for LLMs
Benefits for EAs
Interdisciplinary benefits

**Fig. 3.** Conceptual parallels between large language models and evolutionary algorithms: inspiring novel ideas and technical advancements.

continuously exploring new knowledge. For instance, recent studies have demonstrated that integrating EAs with LLMs can markedly enhance their performance and expand their application scope [15,31,33,40,41,51,52,69,70]. However, a unified paradigm with one-to-one key feature correspondence has not emerged. We stress that the analogy aims not to validate these parallels mathematically but to provide researchers with a pathway to enhance current technical studies. In existing efforts to integrate EAs and LLMs, evolutionary fine-tuning in black-box scenarios and LLM-enhanced EAs are receiving increasing attention. Next, this paper provides a comprehensive review of them to identify key challenges.

## Evolutionary Fine-Tuning in Black-Box Scenarios

The fine-tuning reduces the risk of data leakage and avoids the huge computational cost of training a model from scratch [14]. EA is widely used to fine-tune LLMs in complex scenarios due to their flexibility. Evolutionary model tuning adjusts the model's weights or architecture [72,75,76], requiring a deep understanding of LLM internals. However, real-world constraints like limited computing resources and access restrictions can hinder this process. In contrast, evolutionary prompt tuning [15] and evolutionary self-tuning [77–81] primarily focus on modifying

**Table 2.** A comprehensive summary of evolutionary prompt tuning, highlighting its key characteristics: decision variable and its traits, objective and its traits, adopted method, introducing new models, retraining, and internal access

| Literature | Decision variable | Variable traits | Objective | Objective traits | Adopted methods | New model | Retraining | Internal access |
|---|---|---|---|---|---|---|---|---|
| BBT [15] | Prompt embedding | Continuous | Loss | Single-objective | Random embedding, CMAES | No | No | No |
| BBTv2 [82] | Prompt embedding | Continuous | Loss | Single-objective | Divide-and-conquer, random embedding, CMAES | No | No | Yes |
| Textual inversion [112] | Prompt embedding | Continuous | Loss | Single-objective | Subspace decomposition, random embedding, CMAES | No | No | No |
| SNPE/ABC-SMC [85] | Prompt embedding | Continuous | Loss | Single-objective | Variational inference, random embedding, CMAES | No | No | No |
| PCT [83] | Prompt embedding | Continuous | Loss | Single-objective | Prompt-calibrated tuning, whole-word mask, CMAES | No | No | Yes |
| BBT-RGB [86] | Prompt embedding | Continuous | Loss | Single-objective | Divide-and-conquer, random embedding, COBYLA, CMAES | No | No | Yes |
| BSL [84] | Prompt embedding | Continuous | Loss | Single-objective | Subspace learning, random embedding, CMAES | No | No | No |
| GDFO [87] | Prompt embedding | Continuous | Loss | Single-objective | Knowledge distillation, random embedding, CMAES | Student model | Yes | No |
| FedBPT [88] | Prompt embedding | Continuous | Multi-client loss | Single-objective | Federated CMAES | No | No | No |
| GAP3 [40] | Prompt | Discrete | Performance score, predicted probability | Multi-objective | Multi-level evaluation, genetic algorithm | No | No | No |
| GrIPS [89] | Prompt | Discrete | Accuracy, entropy | Multi-objective | Weighted sum, genetic algorithm | No | No | No |
| ClaPS [90] | Prompt | Discrete | Loss | Single-objective | Clustering and pruning, evolutionary algorithm | No | No | No |
| Attacks [91] | Prompt | Discrete | Cosine similarity | Single-objective | Fitness approximation, genetic algorithm | No | No | No |
| BPT-VLM [92] | Text-image prompt embedding | Continuous | Loss | Single-objective | Random embedding, MMES, MAES, CMAES | No | No | No |

the model's input to enhance performance on specific tasks, requiring access to no internal information. These evolutionary fine-tuning techniques in black-box scenarios are gaining attention for their low cost, as detailed in Tables 2 and 3.

As shown in Fig. 4, evolutionary prompt tuning enhances model generation quality in few-shot or zero-shot settings by searching input prompts. EAs are employed to find prompts to maximize task performance [15], relying solely on LLM inference results. Current approaches are categorized as continuous and discrete prompt tuning. Continuous prompt tuning uses continuous EAs like CMAES to refine prompt embeddings. To enrich the information within the embedding space, various

**Table 3.** A comprehensive summary of evolutionary self-tuning, highlighting its key characteristics: decision variable and its traits, objective and its traits, adopted method, introducing new models, retraining, and internal access

| Literature | Decision variable | Variable traits | Objective | Objective traits | Adopted methods | New model | Retraining | Internal access |
|---|---|---|---|---|---|---|---|---|
| iPrompt [77] | Prompt | Discrete | Render function | Single-objective | LLM-based genetic operators, rank-based selection, exploration | No | No | No |
| Promptbreeder [78] | Task-mutation prompt | Discrete | Performance score | Single-objective | LLM-based genetic operators, genetic algorithm | No | No | No |
| Auto-Instruct [81] | Instruction | Discrete | Predicted score | Single-objective | LLM-based genetic operators, rank-based selection model | Selection model | Yes | No |
| SPELL [79] | Prompt | Discrete | Classification accuracy | Single-objective | LLM-based genetic operators, genetic algorithm | No | No | No |
| EVOPROMPT [80] | Prompt | Discrete | Performance score | Single-objective | LLM-based genetic operators, genetic algorithm, differential evolution | No | No | No |

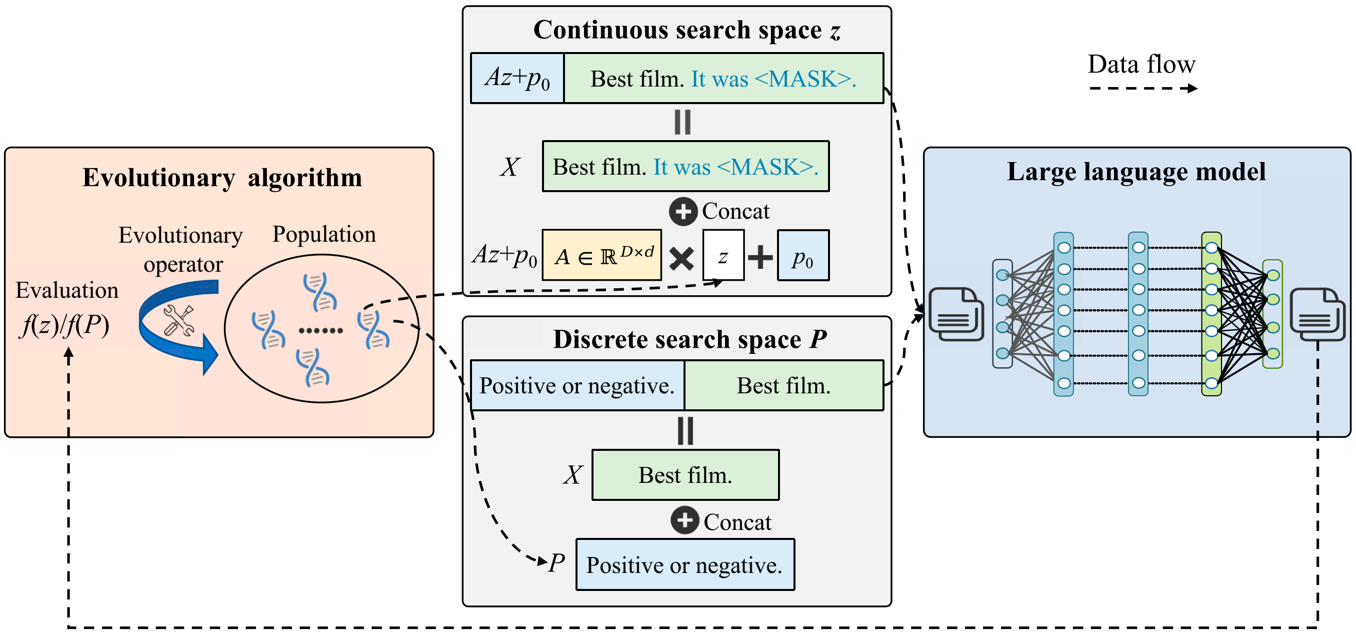


**Fig. 4.** Basic workflow of evolutionary prompt tuning. Evolutionary algorithms are utilized to efficiently search for optimal discrete prompts or continuous prompt embeddings, thereby boosting the performance of large language models on downstream tasks.

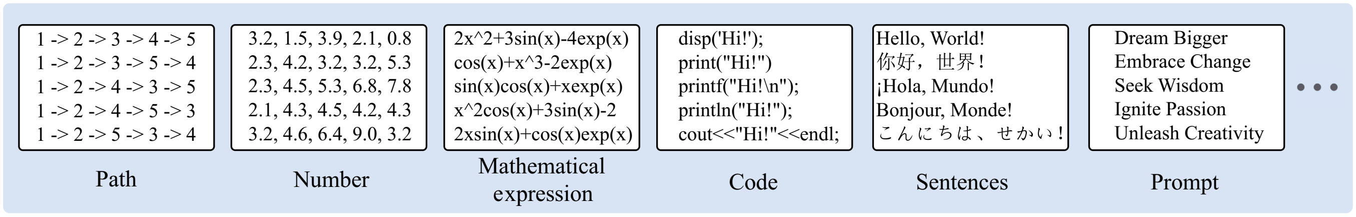
decomposition strategies are incorporated such as divide-and-conquer, subspace learning, and others [23,82–84]. Meanwhile, techniques like knowledge distillation, variational inference, and federated learning are used to boost search efficiency, improve generalization, and enhance security [85–88]. Continuous prompts require embedding space access, unsuitable for strict black-box settings. Discrete prompt tuning directly searches the prompt space using discrete EAs, in which custom genetic operators heuristically modify prompts [40,89]. Zhou et al. [90] clustered and pruned the discrete search space to target promising prompt regions, addressing combinatorial explosion. In addition, evolutionary prompt tuning is also used in adversarial attacks and multi-modal learning [91,92], generating effective attacks and diverse prompts. Recently, LLMs, with strong generative capacity, act as genetic operators in EAs, creating high-quality prompts [77–80], termed self-tuning in this paper. In addition to prompt generation, LLMs can serve as versatile prompt selectors for out-of-domain tasks [81]. Self-tuning works in a flexible language space, independent of parameter updates.

Evolutionary model tuning targets parameter space, while evolutionary prompt tuning and self-tuning focus on language (search) space. Compared to model training involving multiple gradient descents, evolutionary black-box tuning is highly cost-effective. Current research focuses on model evolution within the language space. In open environments, complex tasks may require self-coevolution in language and parameter spaces, posing challenges like resource management, catastrophic forgetting, fitness assessment, and security issues. Efficient resource management strategies help save costs in continuous evolution. Finding a balance between new and old knowledge mitigates catastrophic forgetting. Designing collaborative evaluation strategies for language and parameter spaces tailored to specific tasks is essential. As self-evolving systems continue to advance, the development of robust security assessment mechanisms may become critical to address potential ethical challenges. In addition, integrating text, images, audio, and video is increasingly crucial [93]. Evolutionary multimodal fusion techniques [76,92] offer a promising path to unifying diverse information, thereby expanding the applicability and versatility of evolutionary fine-tuning.

## LLM-Enhanced EA

Figure 5 illustrates how complex individual representations are represented via flexible natural language. Language-represented populations can be directly processed by LLMs with strong text comprehension and generation skills. Table 4 summarizes the LLM-enhanced EAs, where LLMs are employed as the reproduction and mutation operator. These methods maintain the population via LLM-based evolutionary operators to find diverse solutions to complex real-world challenges.

LLM-based reproduction enables the LLMs to derive offspring from parents based on prompts. Prompts usually consist of a problem description (optional), parent population, and task instructions. LLMs apply task instructions to the parent population, generating offspring [41]. Romera-Paredes et al.


1 -> 2 -> 3 -> 4 -> 5
1 -> 2 -> 3 -> 5 -> 4
1 -> 2 -> 4 -> 3 -> 5
1 -> 2 -> 4 -> 5 -> 3
1 -> 2 -> 5 -> 3 -> 4

Path

3.2, 1.5, 3.9, 2.1, 0.8
2.3, 4.2, 3.2, 3.2, 5.3
2.3, 4.5, 5.3, 6.8, 7.8
2.1, 4.3, 4.5, 4.2, 4.3
3.2, 4.6, 6.4, 9.0, 3.2

Number

2x^2+3sin(x)-4exp(x)
cos(x)+x^3-2exp(x)
sin(x)cos(x)+xexp(x)
x^2cos(x)+3sin(x)-2
2xsin(x)+cos(x)exp(x)

Mathematical expression

```
disp('Hi!');
print("Hi!")
printf("Hi!\n");
println("Hi!");
cout<<"Hi!"<<endl;
```

Code

Hello, World!
你好，世界！
¡Hola, Mundo!
Bonjour, Monde!
こんにちは、せかい！

Sentences

Dream Bigger
Embrace Change
Seek Wisdom
Ignite Passion
Unleash Creativity

Prompt

• • •

**Fig. 5.** Various complex individual representations can be represented directly using natural language descriptions, such as paths, numbers, mathematical expressions, code, sentences, and prompts.

**Table 4.** Large language models-enhanced evolutionary algorithms. Large language models are employed as the reproduction and mutation operators in evolutionary algorithms.

| | Reproduction | Mutation |
|---|---|---|
| Prompt construction | Problem description (optional), population, task instructions | Individual, task instructions |
| Operation space | Language space | Language space, parameter space |
| Evolutionary algorithms | Hill climbing, genetic algorithm, genetic programming, quality diversity, MOEA/D, local search | Genetic algorithm, genetic programming, quality diversity |
| Applications | Function search [9], combinatorial optimization [94], reward function optimization [95], automatic machine learning [106–110], multi-objective optimization [96], prompt turning [97], text generation [98,99], game design [100,101], materials science [102], image generation [41], algorithm design [103] | Code generation [32,33], data management [104], fuzzing [105] |

[9] introduced a program search method for mathematical reasoning, where LLMs create multiple programs from parents. Fitness, expressed via numerical values, training logs, and human feedback, is also integrated into prompts to guide the reproduction process [32,33,94–105]. For example, in automated learning, training logs act as fitness for finding efficient architectures and hyperparameters [106–110]. Reproduction using LLMs operates directly in language space, without needing extensive parameter access, resulting in cost savings.

LLMs can also be viewed as mutation operators affecting a single individual. Mutation prompts typically include individual and task instructions. For instance, in the data management strategy SEED [105], LLMs branch an initial code fragment into multiple new code fragments based on task instructions. In addition, Lehman et al. [33] introduced a mutation operator diff that acts on the parameter space. The diff model performs parameter updates in an autoregressive manner, learning incremental changes to files. Given a parent code, diff can simulate the modification behavior of human programmers to generate new code.

Current methods primarily operate in the language space, offering high flexibility and low cost. However, when model parameters are accessible, designing efficient genetic operators in both parameter and language spaces deserves deeper investigation for potential improvements. During evolution, LLMs must address the exploration–exploitation challenge. Exploration encourages the generation of novel and diverse outputs, while exploitation prioritizes outputs that are highly relevant to the given context, potentially sacrificing creativity. Striking a balance between these 2 strategies determines the ability of LLMs to autonomously acquire new knowledge. Evolutionary multi-objective optimization [111] promises to provide a set of solutions with different trade-offs.

## Conclusion

LLMs and EAs have spurred innovation across interdisciplinary domains, with their synergistic integration holding the potential to realize the evolutionary learning machines envisioned by Turing [36]. This paper demonstrates the conceptual parallels between LLMs and EAs from 5 aspects, initially indicating that analogies can potentially spark a new artificial intelligence paradigm integrating LLM's learning abilities with EA's exploratory capabilities. Recently, LLMs have shown promise in utilizing principles of evolution [9,33,41,77–81,95]. The exponential growth in computing power enables large models combined with evolutionary mechanisms to perform reasoning in complex environments. Building on these developments, our work highlights promising directions for advancing current research and identifies critical challenges for future progress.

## Acknowledgments

**Competing interests:** The authors declare that they have no competing interests.

## References

1. Hirschberg J, Manning CD. Advances in natural language processing. *Science*. 2015;349(6245):261–266.
2. Li J, Chiu B, Shang S, Shao L. Neural text segmentation and its application to sentiment analysis. *IEEE Trans Knowl Data Eng*. 2022;34(2):828–842.
3. Li J, Shang S, Chen L. Domain generalization for named entity boundary detection via metalearning. *IEEE Trans Neural Netw Learn Syst*. 2021;32(9):3819–3830.
4. Li J, Han P, Ren X, Hu J, Chen L, Shang S. Sequence labeling with meta-learning. *IEEE Trans Knowl Data Eng*. 2023;35(3):3072–3086.
5. Li J, Feng S, Chiu B. Few-shot relation extraction with dual graph neural network interaction. *IEEE Trans Neural Netw Learn Syst*. 2024;35(10):14396–14408.
6. Radford A, Narasimhan K, Salimans T, Sutskever I. Improving language understanding by generative pre-training. 2018. https://openai.com/research/language-unsupervised
7. Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Burstein J, Doran C, Solorio T, editors. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Minneapolis (MN): Association for Computational Linguistics; 2019. Vol. 1, p. 4171–4186.
8. Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, Neelakantan A, Shyam P, Sastry G, Askell A, et al. Language 500 models are few-shot learners. *Adv Neural Inf Process Syst*. 2020;33:1877–1901.
9. Romera-Paredes B, Barekatain M, Novikov A, Balog M, Kumar MP, Dupont E, Ruiz FJ, Ellenberg JS, Wang P, Fawzi O, et al. Mathematical discoveries from program search with large language models. *Nature*. 2024;625:468–475.
10. Shanahan M, McDonell K, Reynolds L. Role play with large language models. *Nature*. 2023;623:493–498.
11. Schramowski P, Turan C, Andersen N, Rothkopf CA, Kersting K. Large pre-trained language models contain human-like biases of what is right and wrong to do. *Nat Mach Intell*. 2022;4(3):258–268.
12. Liu S, Nie W, Wang C, Lu J, Qiao Z, Liu L, Tang J, Xiao C, Anandkumar A. Multi-modal molecule structure–text model for text-based retrieval and editing. *Nat Mach Intell*. 2023;5(12):1447–1457.
13. Boiko DA, MacKnight R, Kline B, Gomes G. Autonomous chemical research with large language models. *Nature*. 2023;624(7992):570–578.
14. Zheng H, Shen L, Tang A, Luo Y, Hu H, Du B, Tao D. Learn from model beyond fine-tuning: A survey. arXiv. 2023. https://doi.org/10.48550/arXiv.2310.08184
15. Sun T, Shao Y, Qian H, Huang X, Qiu X. Black-box tuning for language model-as-a-service. Paper presented at: International Conference on Machine Learning; 2022; Baltimore, MD, USA.
16. Eiben AE, Smith J. From evolutionary computation to the evolution of things. *Nature*. 2015;7553:476–482.
17. Kudela J. A critical problem in benchmarking and analysis of evolutionary computation methods. *Nat Mach Intell*. 2022;4(12):1238–1245.
18. Jiao L, Zhao J, Wang C, Liu X, Liu F, Li L, Shang R, Li Y, Ma W, Yang S. Nature-inspired intelligent computing: A comprehensive survey. *Research*. 2024;7:0442.
19. Jin Y, Wang H, Sun C. *Data-driven evolutionary optimization*. Cham (Switzerland): Springer; 2021.
20. Miikkulainen R, Forrest S. A biological perspective on evolutionary computation. *Nat Mach Intell*. 2021;3(1):9–15.
21. Stanley KO, Clune J, Lehman J, Miikkulainen R. Designing neural networks through neuroevolution. *Nat Mach Intell*. 2019;1(1):24–35.

22. Matthews D, Spielberg A, Rus D, Kriegman S, Bongard J. Efficient automatic design of robots. *Proc Natl Acad Sci USA*. 2023;120(41):2305180120.
23. Shu JI, Wang Y, Brown A, Kaminsky A. Genetic-algorithm-guided development of parametric aeroelastic reduced-order models with state-consistence enforcement. *AIAA J*. 2023;61(9).
24. Jin Y, Wang H, Chugh T, Guo D, Miettinen K. Data-driven evolutionary optimization: An overview and case studies. *IEEE Trans Evol Comput*. 2018;23(3):442–458.
25. Li B, Wei Z, Wu J, Yu S, Zhang T, Zhu C, Zheng D, Guo W, Zhao C, Zhang J. Machine learning-enabled globally guaranteed evolutionary computation. *Nat Mach Intell*. 2023;5:457–467.
26. Lin T, Chen S, Basu R, Pei D, Cheng X, Kara LB. Target specific peptide design using latent space approximate trajectory collector. arXiv. 2023. https://doi.org/10.48550/arXiv.2302.01435
27. Gupta A, Ong Y-S, Feng L. Insights on transfer optimization: Because experience is the best teacher. *IEEE Trans Emerg Top Comput Intell*. 2018;2(1):51–64.
28. Xue X, Yang C, Feng L, Zhang K, Song L, Tan KC. A scalable test problem generator for sequential transfer optimization. arXiv. 2023. https://doi.org/10.48550/arXiv.2304.08503
29. Gupta A, Ong Y-S, Feng L. Multifactorial evolution: Toward evolutionary multitasking. *IEEE Trans Evol Comput*. 2016;20(3):343–357.
30. Wang C, Liu J, Wu K, Wu Z. Solving multitask optimization problems with adaptive knowledge transfer via anomaly detection. *IEEE Trans Evol Comput*. 2022;26(2):304–318.
31. Hong H, Jiang M. Pre-evolved model for complex multi-objective optimization problems. arXiv. 2023. https://doi.org/10.48550/arXiv.2312.06125
32. Brownlee AE, Callan J, Even-Mendoza K, Geiger A, Hanna C, Petke J, Sarro F, Sobania D. Enhancing genetic improvement mutations using large language models. In: *International symposium on search based software engineering*. Cham: Springer; 2023. p. 153–159.
33. Lehman J, Gordon J, Jain S, Ndousse K, Yeh C, Stanley KO. In: Banzhaf W, Machado P, Zhang M, editors. Evolution through large models. Singapore: Springer; 2024. p. 331–366.
34. Goldberg DE. *Genetic algorithms*. Chennai (India): Pearson Education India; 1989.
35. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I. Attention is all you need. *Adv Neural Inf Process Syst*. 2017;30.
36. Turing AM. *Computing machinery and intelligence*. Dordrecht: Springer; 2009. Vol. 605.
37. Liu J, Sarker R, Elsayed S, Essam D, Siswanto N. Large-scale evolutionary optimization: A review and comparative study. *Swarm Evolution Comput*. 2024;85:101466.
38. Qian H, Yu, Y. Solving high-dimensional multi-objective optimization problems with low effective dimensions. Paper presented at: Proceedings of the AAAI Conference on Artificial Intelligence; 2017; San Francisco, CA, USA.
39. Tian Y, Si L, Zhang X, Cheng R, He C, Tan KC, Jin Y. Evolutionary large-scale multi-objective optimization: A survey. *ACM Comput Surv*. 2021;54(8):1–34.
40. Zhao J, Wang Z, Yang F. Genetic prompt search via exploiting language model probabilities. Paper presented at: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence; 2023; Macao, China.
41. Meyerson E, Nelson MJ, Bradley H, Moradi A, Hoover AK, Lehman J. Language model crossover: Variation through few-shot prompting. arXiv. 2023. https://doi.org/10.48550/arXiv.2302.12170
42. Lyu S, Wu X, Li J, Chen Q, Chen H. Do models learn the directionality of relations? A new evaluation: Relation direction recognition. *IEEE Trans Emerg Top Comput Intell*. 2022;6(4):883–892.
43. Wierstra D, Schaul T, Glasmachers T, Sun Y, Peters J, Schmidhuber J. Natural evolution strategies. *J Mach Learn Res*. 2014;15(1):949–980.
44. Hansen N. The cma evolution strategy: A tutorial. arXiv. 2016. https://doi.org/10.48550/arXiv.1604.00772
45. Voita E, Talbot D, Moiseev F, Sennrich R, Titov I. Analyzing multihead self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In: Korhonen A, Traum D, Marquez L, editors. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence (Italy): Association for Computational Linguistics; 2019.
46. Correia GM, Niculae V, Martins AFT. Adaptively sparse transformers. In: Inui K, Jiang J, Ng V, Wan X, editors. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong (China): Association for Computational Linguistics; 2019.
47. Bhojanapalli S, Yun C, Rawat AS, Reddi S, Kumar S. Low-rank bottleneck in multi-head attention models. Paper presented at: Proceedings of the 37th International Conference on Machine Learning; 2020.
48. Niu Z, Zhong G, Yu H. A review on the attention mechanism of deep learning. *Neurocomputing*. 2021;452:48–62.
49. Deb K, Agrawal RB, Simulated binary crossover for continuous search space. *Complex Syst*. 1995;9(2):115–148.
50. Simon D. *Evolutionary optimization algorithms*. Hoboken (NJ): John Wiley & Sons; 2013.
51. Zhang J, Xu C, Li J, Chen W, Wang Y, Tai Y, Chen S, Wang C, Huang F, Liu Y. Analogous to evolutionary algorithm: Designing a unified sequence model. *Adv Neural Inf Process Syst*. 2021;34:26674–26688.
52. Li X, Wu K, Zhang X, Wang H, Liu J. B2opt: Learning to optimize blackbox optimization with little budget. arXiv. 2023. https://doi.org/10.48550/arXiv.2304.11787
53. Dong Y, Cordonnier J-B, Loukas A. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. Paper presented at: International Conference on Machine Learning; 2021.
54. Zhou Z-H, Yu Y, Qian C. *Evolutionary learning: Advances in theories and algorithms*. Singapore: Springer; 2019.
55. Hassanat A, Almohammadi K, Alkafaween E, Abunawas E, Hammouri A, Prasath VS. Choosing mutation and crossover ratios for genetic algorithms—A review with a new dynamic approach. *Information*. 2019;10(12):390.
56. Su J, Ahmed M, Lu Y, Pan S, Bo W, Liu Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*. 2024;568:127063.
57. Yan H, Deng B, Li X, Qiu X. Tener: Adapting transformer encoder for named entity recognition. arXiv. 2019. https://doi.org/10.48550/arXiv.1911.04474
58. Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W, Liu PJ. Exploring the limits of transfer learning

with a unified text-to text transformer. *J Mach Learn Res*. 2020;21(1):5485–5551.
59. Dai Z, Yang Z, Yang Y, Carbonell J, Le Q, Salakhutdinov R. Transformer-XL: Attentive language models beyond a fixed-length context. In: Korhonen A, Traum D, Marquez L, editors. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence (Italy): Association for Computational Linguistics; 2019.
60. He P, Liu X, Gao J, Chen W. Deberta: Decoding-enhanced bert with disentangled attention. Paper presented at: International Conference on Learning Representations; 2021; Vienna, Austria.
61. Das S, Suganthan PN. Differential evolution: A survey of the state-of-the-art. *IEEE Trans Evol Comput*. 2010;15(1):4–31.
62. Salimans T, Ho J, Chen X, Sidor S, Sutskever I. Evolution strategies as a scalable alternative to reinforcement learning. arXiv. 2017. https://doi.org/10.48550/arXiv.1703.03864
63. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput*. 2002;6(2):182–197.
64. Wagner A, Mitra T, Iyer M, Da Costa G, Tremblay, M.: Position masking for language models. arXiv. 2020. https://doi.org/10.48550/arXiv.2006.05676
65. Singh G, Deb K. Comparison of multi-modal optimization algorithms based on evolutionary algorithms. Paper presented at: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation. GECCO '06; 2006; New York, NY, USA.
66. Das S, Maity S, Qu B-Y, Suganthan PN. Real-parameter evolutionary multimodal optimization—A survey of the state-of-the-art. *Swarm Evolution Comput*. 2011;1(2):71–88.
67. Du Y, Watkins O, Wang Z, Colas C, Darrell T, Abbeel P, Gupta A, Andreas J. Guiding pretraining in reinforcement learning with large language models. Paper presented at: International Conference on Machine Learning; 2023; Honolulu, HI, USA.
68. Burke EK, Gendreau M, Hyde M, Kendall G, Ochoa G, Ozcan E, Rong R. Hyper-heuristics: A survey of the state of the art. *J Oper Res Soc*. 2013;64:1695–1724.
69. Lange R, Schaul T, Chen Y, Lu C, Zahavy T, Dalibard V, Flennerhag S. Discovering attention-based genetic algorithms via meta-black-box optimization. Paper presented at: Proceedings of the Genetic and Evolutionary Computation Conference; 2023; Lisbon, Portugal.
70. Lange RT, Schaul T, Chen Y, Zahavy T, Dalibard V, Lu C, Singh, S, Flennerhag S. Discovering evolution strategies via meta-black-box optimization. Paper presented at: The Eleventh International Conference on Learning Representations; 2023; Kigali, Rwanda.
71. Ding L, Zhang J, Clune J, Spector L, Lehman J. Quality diversity through human feedback: Towards open-ended diversity-driven optimization. Paper presented at: Forty First International Conference on Machine Learning; 2024; Vienna, Austria.
72. Klein A, Golebiowski J, Ma X, Perrone V, Archambeau C. Structural pruning of large language models via neural architecture search. Paper presented at: AutoML Conference 2023 (Workshop); 2023; Potsdam/Berlin, Germany.
73. Sener O, Koltun V. Multi-task learning as multi-objective optimization. *Adv Neural Inf Process Syst*. 2018;31.
74. Lin X, Zhen H-L, Li Z, Zhang Q-F, Kwong S. Pareto multi-task learning. *Adv Neural Inf Process Syst*. 2019;32.
75. Choong HX, Ong Y-S, Gupta A, Chen C, Lim R. Jack and masters of all trades: One-pass learning sets of model sets from large pre-trained models. *IEEE Comput Intell Mag*. 2023;18(3):29–40.
76. Du G, Li J, Liu H, Jiang R, Yu S, Guo Y, Goh SK, Tang H-K. Knowledge fusion by evolving weights of language models. arXiv. 2024. https://doi.org/10.48550/arXiv.2406.12208
77. Singh C, Morris JX, Aneja J, Rush A, Gao J. Explaining data patterns in natural language with language models. In: Belinkov Y, Hao S, Jumelet J, Kim N, McCarthy A, Mohebbi H, editors. *Proceedings of the 6th Blackbox NLP Workshop: Analyzing and Interpreting Neural Networks for NLP*. Singapore: Association for Computational Linguistics; 2023.
78. Fernando C, Banarse D, Michalewski H, Osindero S, Rocktaschel T. Promptbreeder: Self-referential self-improvement via prompt evolution. arXiv. 2023. https://doi.org/10.48550/arXiv.2309.16797
79. Li YB, Wu K. Spell: Semantic prompt evolution based on a llm. arXiv. 2023. https://doi.org/10.48550/arXiv.2310.01260
80. Chen A, Dohan D, So D. Evoprompting: Language models for code-level neural architecture search. Paper presented at: Thirty-Seventh Conference on Neural Information Processing Systems; 2023; New Orleans, LA, USA.
81. Zhang Z, Wang S, Yu W, Xu Y, Iter D, Zeng Q, Liu Y, Zhu C, Jiang M. Auto-instruct: Automatic instruction generation and ranking for black-box language models. In: Bouamor H, Pino J, Bali K, editors. *Findings of the association for computational linguistics: EMNLP 2023*. Singapore: Association for Computational Linguistics; 2023. p. 9850–9867.
82. Sun T, He Z, Qian H, Zhou Y, Huang X-J, Qiu X. Bbtv2: Towards a gradient-free future with large language models. Paper presented at: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing; 2022; Abu Dhabi, United Arab Emirates.
83. Qi S, Zhang Y. Prompt-calibrated tuning: Improving black-box optimization for few-shot scenarios. Paper presented at: 2023 4th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT); 2023; Nanjing, China.
84. Zheng Y, Tan Z, Li P, Liu Y. Black-box prompt tuning with subspace learning. *IEEE/ACM Trans Audio Speech Lang Process*. 2024;32:3002–3013.
85. Shen M, Ghosh S, Sattigeri P, Das S, Bu Y, Wornell G. Reliable gradient-free and likelihood-free prompt tuning. In: Vlachos A, Augenstein I, editors. *Findings of the Association for Computational Linguistics: EACL 2023*. Dubrovnik (Croatia): Association for Computational Linguistics; 2023. p. 2416–2429.
86. Sun Q, Han C, Chen N, Zhu R, Gong J, Li X, Gao M. Make prompt based black-box tuning colorful: Boosting model generalization from three orthogonal perspectives. Paper presented at: Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024) ELRA and ICCL; 2024; Torino, Italia.
87. Han C, Cui L, Zhu R, Wang J, Chen N, Sun Q, Li X, Gao M. When gradient descent meets derivative-free optimization: A match made in black-box scenario. In: Rogers A, Boyd-Graber J, Okazaki N, editors. *Findings of the Association for Computational Linguistics: ACL 2023*. Toronto (Canada): Association for Computational Linguistics; 2023. p. 868–880.

88. Sun J, Xu Z, Yin H, Yang D, Xu D, Chen Y, Roth HR. Fedbpt: Efficient federated black-box prompt tuning for large language models. arXiv. 2023. https://doi.org/10.48550/arXiv.2310.01467.
89. Prasad A, Hase P, Zhou X, Bansal M. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In: Vlachos A, Augenstein I, editors. *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Dubrovnik (Croatia): Association for Computational Linguistics; 2023.
90. Zhou H, Wan X, Vulić, I, Korhonen, A. Survival of the most influential prompts: Efficient black-box prompt search via clustering and pruning. In: Bouamor H, Pino J, Bali K, editors. *Findings of the Association for Computational Linguistics: EMNLP 2023*. Singapore: Association for Computational Linguistics; 2023. p. 13064–13077.
91. Lapid R, Langberg R, Sipper M. Open sesame! Universal black box jailbreaking of large language models. arXiv. 2023. https://doi.org/10.48550/arXiv.2309.01446
92. Yu L, Chen Q, Lin J, He L. Black-box prompt tuning for vision-language model as a service. Paper presented at: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence; 2023; Macao, China.
93. Fei N, Lu Z, Gao Y, Yang G, Huo Y, Wen J, Lu H, Song R, Gao X, Xiang T, et al. Towards artificial general intelligence via a multimodal foundation model. *Nat Commun*. 2022;13(1):3094.
94. Liu S, Chen C, Qu X, Tang K, Ong YS. Large language models as evolutionary optimizers. arXiv. 2023. https://doi.org/10.48550/arXiv.2310.19046
95. Ma YJ, Liang W, Wang G, Huang D-A, Bastani O, Jayaraman D, Zhu Y, Fan L, Anandkumar A. Eureka: Human-level reward design via coding large language models. Paper presented at: The Twelfth International Conference on Learning Representation; 2024; Vienna, Austria.
96. Liu F, Lin X, Wang Z, Yao S, Tong X, Yuan M, Zhang Q. Large language model for multi-objective evolutionary optimization. arXiv. 2023. https://doi.org/10.48550/arXiv.2310.12541
97. Yang C, Wang X, Lu Y, Liu H, Le QV, Zhou D, Chen X. Large language models as optimizers. Paper presented at: The Twelfth International Conference on Learning Representations; 2024; Vienna, Austria.
98. Xiao L, Chen X. Enhancing llm with evolutionary fine tuning for news summary generation. arXiv. 2023. https://doi.org/10.48550/arXiv.2307.02839.
99. Bradley H, Dai A, Teufel H, Zhang J, Oostermeijer K, Bellagente M, Clune J, Stanley K, Schott G, Lehman J. J Quality-diversity through ai feedback. arXiv. 2023. https://doi.org/10.48550/arXiv.2310.13032.
100. Lanzi PL, Loiacono D. Chatgpt and other large language models as evolutionary engines for online interactive collaborative game design. Paper presented at: Proceedings of the Genetic and Evolutionary Computation Conference. GECCO '23; 2023; New York, NY, USA.
101. Sudhakaran S, González-Duque M, Freiberger M, Glanois C, Najarro E, Risi S. MarioGPT: Open-ended text2level generation through large language models. Paper presented at: Thirty-Seventh Conference on Neural Information Processing Systems; 2023; New Orleans, LA, USA.
102. Jablonka KM, Ai Q, Al-Feghali A, Badhwar S, Bocarsly JD, Bran AM, Bringuier S, Brinson LC, Choudhary K, Circi D, et al. 14 examples of how llms can transform materials science and chemistry: A reflection on a large language model hackathon. *Dig Discov*. 2023;2(5):1233–1250.
103. Liu F, Tong X, Yuan M, Lin X, Luo F, Wang Z, Lu Z, Zhang Q. An example of evolutionary computation+ large language model beating human: Design of efficient guided local search. arXiv. 2024. https://doi.org/10.48550/arXiv.2401.02051.
104. Chen Z, Cao L, Madden S, Fan J, Tang N, Gu Z, Shang Z, Liu C, Cafarella M, Kraska T. Seed: Simple, efficient, and effective data management via large language models. arXiv. 2023. https://doi.org/10.48550/arXiv.2310.00749.
105. Xia CS, Paltenghi M, Tian JL, Pradel M, Zhang L. Fuzz4all: Universal fuzzing with large language models. Paper presented at: 2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE); 2024; Lisbon, Portugal.
106. Nasir MU, Earle S, Togelius J, James S, Cleghorn C. Llmatic: Neural architecture search via large language models and quality-diversity optimization. arXiv. 2023. https://doi.org/10.48550/arXiv.2306.01102.
107. Zheng M, Su X, You S, Wang F, Qian C, Xu C, Albanie S. Can gpt-4 perform neural architecture search? arXiv. 2023. https://doi.org/10.48550/arXiv.2304.10970.
108. Wang H, Gao Y, Zheng X, Zhang P, Chen H, Bu J. Graph neural architecture search with gpt-4. arXiv. 2023. https://doi.org/10.48550/arXiv.2310.01436.
109. Zhang M, Desai N, Bae J, Lorraine J, Ba J. Using large language models for hyperparameter optimization. Paper presented at: NeurIPS 2023 Foundation Models for Decision Making Workshop; 2023; New Orleans, LA, USA.
110. Zhang S, Gong C, Wu L, Liu X, Zhou M. Automl-gpt: Automatic machine learning with gpt. arXiv. 2023. https://doi.org/10.48550/arXiv.2305.02499.
111. De Ath G, Everson RM, Rahat AAM, Fieldsend JE. Greed is good: Exploration and exploitation trade-offs in Bayesian optimisation. *ACM Trans Evol Learn Optim*. 2021;1(1):1–22.
112. Fei Z, Fan M, Huang J. Gradient-free textual inversion. Paper presented at: Proceedings of the 31st ACM International Conference on Multimedia. MM '23; 2023; New York, NY, USA.