Exploring the Prompt Space of Large Language Models through Evolutionary Sampling

Martina Saletta
martina.saletta@unibg.it
Dept. of Human and Social Sciences
University of Bergamo
Bergamo, Italy

Claudio Ferretti
claudio.ferretti@unimib.it
Dept. of Informatics, Systems and Communication
University of Milano-Bicocca
Milan, Italy

ABSTRACT

Large language models (LLMs) are increasingly gaining relevance in every-day life, due to their apparent ability in solving tasks that demand intricate linguistic comprehension. Recent studies state that one of the key points that impact their outcome is the quality of the prompt used to interact with them. This work proposes a grammar-based evolutionary approach for exploring the prompt space of LLMs, driven by a fitness function that aims at optimizing the performance on a given task. We tested our technique by steering two state-of-the-art models through evolved prompts, and by comparing the performance they obtain on 8 benchmark tasks with that obtained when using other baseline prompts on the same tasks, showing that in most cases our prompts yield better results. Further, we defined a *constrained* mutation operator that limits the changes to specific grammar non-terminals, allowing to study and highlight the elements in the prompt that mostly affect the output of the LLM. Finally, a thorough discussion points out some issues that limit the relevance of the emerging prompt engineering discipline, given the existence of many effective prompt structures and the possible diversity that can be observed in the LLM output given the same input to the model.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Genetic algorithms**; *Neural networks.*

KEYWORDS

Large language models, Prompt evolution, Evolution strategies, Local search

ACM Reference Format:
Martina Saletta and Claudio Ferretti. 2024. Exploring the Prompt Space of Large Language Models through Evolutionary Sampling. In *Genetic and Evolutionary Computation Conference (GECCO '24), July 14–18, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3638529.3654049

GECCO '24, July 14–18, 2024, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0494-9/24/07.
https://doi.org/10.1145/3638529.3654049

1 INTRODUCTION

Large language models (LLM), queried by giving them as input a natural language question, are able to generate textual answers on many different topics and which require reasoning at different levels. Actually, the quality of their answers heavily depends on the topic, on the required reasoning effort, and most importantly on the way the question is structured. In the literature, several prompt engineering techniques have been presented, aiming at enhancing the performance of LLMs on diverse tasks. The search for a good prompt is not easy, due to the internal working of the models, which are trained to learn complex and wide correlations between parts of huge textual training datasets. Also, given an input query, they generate an output answer that is controlled by the learned internal representation, and partly by random noise which by design is part of the computation. Moreover, the search space for a good query prompt is virtually unconstrained, consisting of all natural sentences which can be built around the data about which the LLM is required to reason and answer.

This work introduces a novel way to automatically search for effective prompts based on an evolutionary computation approach, and aiding the exploration of the solution space with constraints derived from known prompt engineering techniques. Even under the constraints we designed, the system is allowed to generate and select novel and successful prompt structures, which are exploiting mixtures and new variations of known querying approaches. The key idea is evolving prompts, our individuals, under the control of custom designed grammars, where a collection of query components and some details of the given task are encoded.

This work presents the following contributions:

- a novel model agnostic system to generate effective prompts,
- an analysis of the results of its application to on two open source large language models when asked to face eight reasoning tasks, reaching better than state of the art scores on some of them,
- experimental evidence of emergent properties of large language systems, in terms of how specific prompt features are differently effective for each model on any given task, producing evidence of how hard could be the search for a definitive and universal prompting technique.

2 RELATED WORK

In very recent times, generative AI models known as Large Language Models, that are based on the transformer architecture [21], appeared to be good in solving several and diverse tasks and attained a notable presence in everyday life, owing to their capacity to generate human-like texts and to the widespread availability

of models (e.g. GPT [1], PaLM [2] and LLaMA [20]). Some well performing models are derived from open sourced foundational models, such as LLaMA or Mistral 7B [5]. Most of these models are made available in different sizes, measured in terms of how many parameters the model uses, that is the quantity of weights and biases of its neural architecture. Usually model size ranges from a few billions up to hundreds of billions parameters. We will operate on general purpose open source LLMs in the 7B size range.

Large language models are evaluated on benchmark datasets, concerning different types of Q&A effort for the LLM, which generates the answers. For instance, Beyond the Imitation Game benchmark [18] (BIG-bench) consists of 204 tasks mostly related to reasoning, while Massive Multitask Language Understanding benchmark [4] (MMLU) offers 57 tasks of different difficulty levels. We will measure performances with respect to 8 reasoning tasks from BIG-bench.

The way a question is presented to a LLM has been shown to impact the correctness of the generated answer. Any prompt has to include the question to answer and the data on which to elaborate the answer. But a few richer human-designed prompt structures have been studied. Usually they add one or more of these parts: one or more examples of data together with the corresponding correct answers [1], a possible suggested contextual role [8, 22], and further recommendations which aim at eliciting the generation of better answers [7]. All these parts affect the system performance based on the internal representations learned by the LLM. Our work proposes a novel evolutionary system generating several combinations of all of these variations, in a controlled way.

In the literature, different attempts to automatically generate effective prompts can be found. We can mention, for instance, AutoPrompt [16], an automated method to create prompts based on a gradient-guided search, and a system based on the idea of maximizing the mutual information between an input and the corresponding model output [17]. The systems APE [24] and EvoPrompt [3] generate and optimize prompts by iteratively modifying and evaluating templates, and templates are initially generated by a LLM (APE) or manually (EvoPrompt). Moreover, in both systems the variations of the prompts are proposed by a LLM. Our approach is different for two main key aspects: the random initial population and the generation of mutated individuals are all guided only by rules detailed in a formal grammar, and the LLM is used only to generate answers, which then determine fitness scores. The technique closely follows the usual evolutionary approach.

3 PROMPT EVOLUTION

This work stands in the setting of the emerging *prompt engineering* discipline, that is the process of writing text for interacting with a generative AI model. The founding idea is that the search of a good prompt can be seen as a problem in which, given a model $\mathcal{M}$ and a pair (Q, A) consisting in a question paired with the corresponding expected answer, we look for an instruction p that, when combined with Q through a function $f(p, Q)$ yields the model $\mathcal{M}$ to answer A when prompted with $f(p, Q)$. More in general, we can describe this as an optimization problem in which, given a set of $D = \{(Q_1, A_1), \ldots, (Q_n, A_n)\}$ pairs of question/answer demonstrations, we look for a single instruction p and for a function f that maximizes the probability to obtain A_i when prompting $\mathcal{M}$ with $f(p, Q_i)$.

This work addresses this problem through an evolutionary approach based on grammatical evolution (GE) [13], which allows to evolve individuals whose phenotypes are compliant with a given formal grammar. Due to the problems of redundancy (lots of genotypes maps to the same phenotype) and locality (a small change in the genotype can cause radical changes in the phenotype) that occur in standard GE [10], we base our experiments on a more recent approach named dynamic structured grammatical evolution (DSGE) [10, 11], which introduces changes in the genome representation so as to face these drawbacks. On top of DSGE, we can design an approach that allows us both to explore the prompt space and to identify the features that mostly affect the performance of a LLM on a given task.

3.1 Two-stage evolutionary sampling

Basically, in DSGE, a genotype is represented by a list in which each position refers to a non-terminal symbol of a given context free grammar, and each gene is a list of ordered derivation steps. Formally, each genotype is a sequence $S_1, \ldots, S_n$. Here, for each $1 \leq i \leq n$, S_i denotes a non-terminal symbol derived from a context-free grammar $G = (N, T, A, P)$. In this grammar, N and T stand for the sets of non-terminal and terminal symbols, respectively, $A \in N$ represents the axiom or starting symbol of G, and P is the set of production rules. Additionally, each S_i is expressed as a list of integers $r_1, \ldots, r_k$, where k is the number of times the i-th non-terminal is expanded. These integer values correspond to the indices (in the grammar) of the applied expansion rules.

Through DSGE, we explored the prompt space through a two-phase evolutionary procedure. In the first one, we applied standard DSGE, with grammar and fitness function as defined in the following, while in the second one we used the best individuals found in the first phase as initial populations for a $(\mu + \lambda)$ evolution strategy [12], that is a population-based evolutionary procedure in which only mutations are allowed (without crossover), and each generation is composed by the μ best individuals of the previous generation, along with λ/μ mutated variants for each of them.

Prompt grammar. With this equipment, we need to define a prompt grammar to effectively explore the space. Since a search in the full space of possible English sentences would be computationally infeasible, to define the grammar that guide the evolution we tap into the recent literature that emerged in the field of prompt engineering. We consider a prompt as a composition of possible parts, and we model them through the grammar in Figure 1. We consider four possible prompt structures, each composed by different elements derived from the literature, that are represented in the grammar by non-terminal symbols:

- **<zero>:** is a base prompt composed by a simple request describing a task, and an input
- **<cot>:** represent a prompt in which a request is augmented with one or more examples [1]
- **<zerocot>:** is a prompt augmented with a sentence that suggest to decompose the reasoning process in different steps [7]

<context>: represents a sentence describing a situation or that suggests the LLM to play a given role, following the idea that this kind of role-playing can enhance its performance [8, 22]

<sbs>: represents the sentence used in the <zerocot> prompt structure

Each of the productions referred to the non-terminals <sbs>, <context> and <req> is a set of 100 terminal symbols, to generate 100 sentences with equivalent semantics but expressed in different manners. To produce these sets, we started from base formulations (i.e. "Let's think step by step", "You are a human that needs to solve a task.", and the base request as expressed in the dataset for a given task, respectively) and we asked ChatGPT to formulate 100 semantically similar variants. The productions referred to the non-terminals <example> are instead sets of 10 examples taken from the dataset. Notice that <context> and <sbs> are generic, while <req> and <example> are specific of the dataset.

```
<prompt> ::= <zero> | <cot> | <zerocot> | <combi>

<zero> ::=  <req> Please, answer with only "yes" or "no".\n<input>
    | <input>\n<req> Please, answer with only "yes" or "no".
    | <context>\n<zero>

<cot> ::= <req> Please, answer with only "yes" or "no".
            For instance:\n<examples>\n<input>
    | <zero> For instance:\n<examples>

<zerocot> ::= <zero>\n<sbs>
    | <sbs>\n<zero>
    | <req> Please, answer with only "yes" or "no".\n<sbs>\n<input>

<combi> ::= <req> Please, answer with only "yes" or "no".
            For instance:\n<examples>\n<sbs><input>
    | Consider these examples:\n<examples>\n<zerocot>
    | <cot>\n<sbs>

<examples> ::= <example>
    | <example>\n<examples>

<input> ::= [[INPUT]]
<context> ::= c1 | ... | c100
<req> ::= r1 | ... | r100
<example> ::= ex1 | ... | ex100
<sbs> ::= s1 | ... | s100
```

Figure 1: Grammar template for evolving prompts. <req> and <example> productions are proper to the task, while <context> and <sbs> are the same in each task.

Fitness function. To measure the quality of the individuals during the evolution, we measured the performance of the LLM under exam on a sample of 100 instances taken from the dataset referred to the given task, when prompted with the evolved template. Such performance is defined as the accuracy score with respect to the expected binary answers. Notice that, in general, this measure is only an heuristic, since to have a more precise measure the performance should be computed on the whole dataset, but this would be computationally too expensive, especially on tasks containing bigger numbers of instances (see Section 4). Also, we are aware that measuring the fitness on a fixed number of instances "favours" the tasks with a small number of instances, since the proportion compared to the total number of instances is higher, thus leading to a more precise estimate, less susceptible to the randomness of the sample. However, this is not a significant issue, given the promising results obtained when testing the evolved prompts on the whole dataset.

Constrained mutation operators. For the second evolutionary phase, we used a "constrained" mutation operator that limits the mutation only to given subsets of non-terminals. In DSGE, standard mutation for a position $S_i = r_1, \ldots, r_k$ is defined as a random change of an integer r_j into another valid integer r_j^* for that gene, corresponding to the selection of a different expansion option for a non-terminal symbol. Following an approach similar to that described in [15], and by considering the constrained mutation operator as defined in the same work, by limiting the mutation only to given non-terminals, and by analysing the fitness variations, we can observe which are the prompt features that mostly affect the output of the LLM. As it will be thoroughly discussed in Section 6, these features vary depending on the task and on the model.

3.2 Prompt evaluation and analysis

The evolutionary search allowed us to obtain a broad range of candidate prompts. Given the limits of the considered fitness function already outlined in the previous paragraph, merely selecting the candidate with the highest fitness would have been restrictive, and could have led to the selection of a non-optimal candidate. To face this issue, we defined a more robust procedure to select the winning prompt template, given a task. This procedure, formally described in Algorithm 1, basically considers all the prompts found during all the generations of the second phase having a fitness higher than the fitness of the individual that resulted the best after the first phase, and selects the one that the most number of times has been evaluated with a good fitness. In summary:

- Among all the individuals evolved in the second phase, consider the subset P having fitness higher than the fitness of the best individual after the first phase;
- List all the unique prompts, and keep trace of the times each prompt occurred, and the worst fitness with which it was evaluated;
- Return the prompt that occurred the most number of times. In case of draw, return that with the highest worst fitness.

4 EXPERIMENTAL SETTING

The outlined prompt evolution framework has been tested by considering the performance of two different open source models on 8 benchmark tasks when queried with the prompts obtained through the evolutionary search, and by comparing it with that obtained with state of the art prompts. We now provide a detailed description of methods and materials used in the experiments, which are also publicly available[1].

4.1 Models

Many state of the art LLMs such as ChatGPT, despite their potential in assisting in several diverse jobs, entail considerable expenses, due to the intensive computational resources required to run them and to the cost per-token for an automated use. For this reason, in this work we prefer to develop the discussion by focusing on smaller open source models, due to their availability and since they are in general easy to use and require relatively low computational resources. This is a crucial point, as our results demonstrate that the

[1] https://github.com/Martisal/evoPrompt

Algorithm 1 Best prompt selection

```
P ← p_1, ..., p_n                                   ▷ initial population
T ← task
minfit ← best fitness obtained after the first evo phase for T
for i ∈ {1, ..., n} do
    if FITNESS(p_i) ≤ minfit then
        remove p_i from P
    end if
end for
n ← |P|
prompts, card, fit ← empty lists
for i ∈ {1, ..., n} do
    if PHENOTYPE(p_i) ∉ prompts then
        append PHENOTYPE(p_i) to prompts
        append c_i = 1 to card
        append f_i =FITNESS(p_i) to fit
    else
        j ← index of PHENOTYPE(p_i) ∈ prompts
        c_j = c_j + 1
        if f_j > FITNESS(p_i) then
            f_j = FITNESS(p_i)
        end if
    end if
end for
bestfit ← 0
for i ∈ {1, ..., |card|} do
    if c_i = MAX(card) then
        if f_i > bestfit then
            bestprompt = p_i
            bestfit = f_i
        end if
    end if
end for
output bestprompt
```

performance of these small models can be significantly enhanced with an optimized prompt. To grant robustness, the proposed automatic prompt optimization technique has been tested on two state-of-the-art lightweight open source models:

Vicuna-7B: (version 1.5), an open source model[2] derived from LLaMA [20] that has been evaluated on a conversational benchmark by using GPT-4 [14] as a judge [23], and that outperformed other open source models such as LLaMA and Alpaca [19] by reaching 90% quality of ChatGPT.

Starling-7B: an open source model[3] derived from Mistral [5] and trained by reinforcement learning [6] with feedback from AI that has been evaluated in MT Bench with GPT-4 as a judge, outperforming every model to date on MT-Bench except for GPT-4 and GPT-4 Turbo.

4.2 Benchmark dataset

The response of the LLMs to different prompt has been evaluated by using the Beyond the Imitation Game benchmark (BIG-bench) [18], which consists of a collection of questions and problems, distributed over 204 tasks belonging to different topics, such as linguistics, reasoning, programming or specific disciplines. In particular, for our discussion we considered 8 reasoning tasks belonging to the BIG-bench Lite suite – a lightweight evaluation set composed of 24 tasks – that required a binary answer:

Causal judgment: answer questions about causal attribution (180 instances)

Epistemic reasoning: determine whether one sentence entails the next (1990 instances)

Hyperbaton: order adjectives correctly in English sentences (≈ 50K instances)

Implicatures: predict whether an answer to a given question counts as a yes or as a no (483 instances)

Logical fallacy detection: detect both informal and formal logical fallacies (2790 instances)

Navigate: given a series of navigation instructions, determine whether one would end up back at the starting point (990 instances)

Snarks: determine which of two sentences is sarcastic (171 instances)

Winowhy: evaluate the reasoning in answering Winograd Schema Challenge [9] questions (2855 instances)

4.3 Baseline prompts

The leading goal of prompt engineering is to study effective ways for building prompts that are able to elicit a better performance or to enhance the *reasoning* capabilities of the LLMs. The literature presents several examples of techniques that can be used to build effective prompt templates. For this work, among the wide variety of prompting techniques, we considered two of the most popular and well-established approaches as baselines:

few-Shot: an interaction strategy that aims at enhancing the quality of the responses by helping the LLM by augmenting the prompt with one or more question/answer pairs as examples [1]. In this work we tested prompts with a single example (1-Shot).

0-Shot-CoT: from here denoted as 0-CoT, a prompt template that induces a chain of thoughts by closing the prompt with the sentence "Let's think step by step" [7].

4.4 Evolutionary search

The evolutionary exploration of the prompt space has been tackled in 2 phases. In both these steps, the fitness function was defined, given a task, as the percentage of correct answers obtained by the model when queried with the prompt on a set of 100 randomly chosen examples taken from the dataset.

In the first phase, through a standard DSGE [10] algorithm we generated populations of individuals compliant with the grammar reported in Figure 1. Notice that, for each task, the grammars differ for the productions that expand the non-terminals `<example>` and `<req>` which are proper to the problem in exam. For this phase, populations of 30 individuals have been evolved for 10 generations with crossover and mutation probabilities of 0.9 and 0.1, respectively, an elite of 3 individuals, and tournament selection with tournament

[2]Available: https://github.com/lm-sys/FastChat (accessed on 25 January 2024)
[3]https://starling.cs.berkeley.edu/

size equals to 5. The results of this preliminary evolutionary phase are reported in Figure 2.

Then, the best individuals that emerged during this evolutionary phase, chosen according to the procedure defined by Algorithm 1, have been used to compose starting populations for a second evolutionary step in which only mutations were allowed, and operators have been constrained so as to allow variations only to specific non-terminals. In particular, four $(\mu+\lambda)$-ES [12] runs have been performed, with $\mu = 10$ and $\lambda = 30$, each having a distinct constraints set for the mutation operator:

(1) mutations allowed only in the gene referred to the non-terminal `<context>`;
(2) mutations allowed only in the gene referred to the non-terminal `<req>`;
(3) mutations allowed only in the gene referred to the non-terminal `<sbs>`;
(4) mutations allowed in the genes referred to the non-terminals `<context>`, `<req>`, `<sbs>` and `<example>`.

The results of this second evolutionary phase are presented in Figure 3, where we reported the variation step among the best individual after the first phase and that obtained by applying ES with mutation constraints.

Finally, for each task, we identified the "winning" prompt by considering all the individuals generated with the ES runs having a fitness higher than the fitness of the best individual found during the first phase, i.e. all the individuals that have undergone an improvement through the local search. Among them, we selected the prompt that occurred the most number of times. In case of equality, we selected the one that has been evaluated better (see the procedure in Algorithm 1). We then evaluated these prompts on the whole set of instances for each task, and compared the results with that obtained when using other baseline prompts. Results are reported in Table 2, while the tested prompts are in Table 3. Also, we considered the union of the same sets of individuals having fitness higher that the best after the first phase, and computed the percentage of prompts having specific syntactic features (Table 1), so as to identify the structures that seem to mostly affect the LLM output.

5 RESULTS

The results of our experiments can be examined from the point of view of the raw performance of selected prompts in the different tasks, together with their comparison against state of the art prompt structures, and from the point of view of the details of the evolutionary dynamics and of the structures of the individuals. It is useful to remark that for the tasks we considered, all taken from BIG-bench, the answers consist of a binary classification, and the score of a machine generating random answers would be 0.50. Also, the fitness function is computed as the score the answers of an individuals on a random set of instances for the task at hand.

Figure 2 shows for both Vicuna and Starling how the fitness scores change for each task during the first generations. In all cases the best fitness is above the random performance since the first generation, while the average fitness in some cases tends to decrease over successive generations. Notable cases are the results for Hyperbaton, the only task where the best fitness is above 0.75

Table 1: Percentage of presence of different prompt elements in the best prompts found through the evolutionary search.

Task	Model	Role	Shots	0-CoT
Causal judgment	Vicuna	0.123	0.398	0.771
	Starling	0.364	0.644	1.000
Epistemic reasoning	Vicuna	0.228	1.000	0.447
	Starling	0.049	1.000	0.958
Hyperbaton	Vicuna	0.000	1.000	0.073
	Starling	0.279	0.721	0.721
Implicatures	Vicuna	0.057	0.329	0.893
	Starling	0.400	1.000	1.000
Logical fallacy detection	Vicuna	0.000	1.000	1.000
	Starling	0.000	0.942	1.000
Navigate	Vicuna	0.451	0.976	0.733
	Starling	0.367	0.917	0.917
Snarks	Vicuna	0.000	1.000	0.527
	Starling	0.082	0.749	0.298
Winowhy	Vicuna	0.000	0.737	0.263
	Starling	0.250	0.750	1.000

and the average stays better than random, and for the Navigate task, where the average fitness is for most of the gernations as low as 0.25. In general, it appears that the exploration of the prompt space, controlled by the options available in the grammar we defined for each task, quickly finds a prompt which leads the LLM to generate answers better than random, but the evolutionary exploitation of the best individuals does not produce individuals much better than the best individuals available at the beginning. It is important to stress that this first evolutionary stage is not actually used to find single optimized solutions, but instead to generate a set of good individuals, which are then improved in the second stage and eventually examined by Algorithm 1, which selects the final optimal prompt structure.

Concerning the second evolutionary phase, where the exploration of alternative prompt structures is more constrained than in the first phase, according to the sets of allowed mutations we defined, we show in Figure 3 the gains obtained for the performance of the prompts selected by the evolution strategy. For all the constraint sets and for most of the tasks, individuals with improved scores have been found. But quantitatively the gains obtained are different from task to task. Considering for instance Vicuna, three out of eight tasks show a major improvement, namely Epistemic reasoning, Hyperbaton, and Implicatures. For the remaining five tasks only slightly improved individuals have been found. For all of them each constraint on the types of mutation, among those available in the grammars, gave different gains with respect to the fitness reached during the first evolutionary stage. We remark that the possible contextual roles in the prompts were the same for all the task grammars.

After the evolutionary exploration of the prompt space, the individual finally selected with Algorithm 1 have been scored on the complete set of instances for each of their respective task. The results are shown in *evo* column of Table 2, together with the performances of the prompt structures directly defined after the recommendations from literature, namely the *base* structure

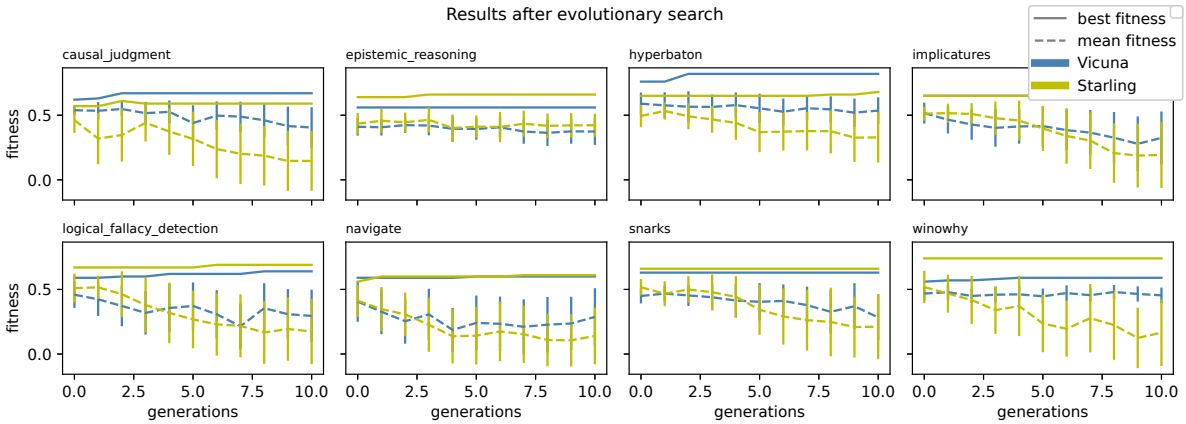

Figure 2: Best and average fitness after 10 generations. Vertical bars represent the variance.

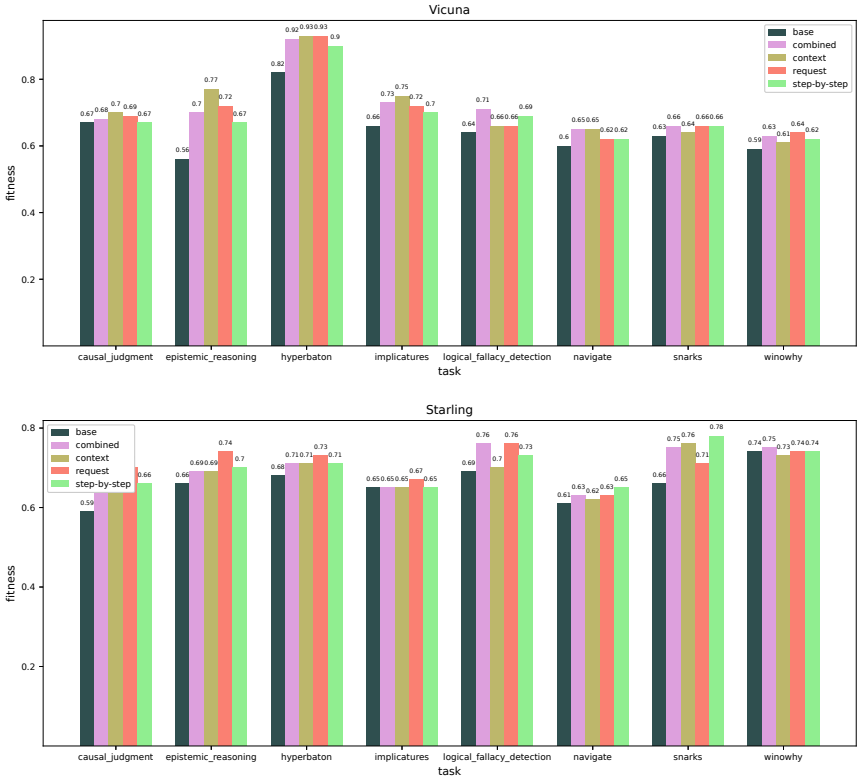

Figure 3: Fitness improvement after ES-$(10 + 30)$. Gray bars represent the best fitness after the evolutionary search, colored bars represent the best fitness after ES with different constraints.

only including the question defined by the task, the *0-CoT* structure with added the "step-by-step" suggestion to the LLM, and the *1-Shot* prompts. Such prompts are directly defined from the question posed by each task, while sharing the same structure for all the benchmark problems. On the other hand, the individuals evolved for the tasks, whose performance is in *evo* column, can have a different structure for each of them, as can be seen in Table 3, where we give the actual text of query templates for each task and each model.

Table 2 also includes as a reference the scores obtained with baselines prompts by the state of the art model PaLM-8B[4], run in the configuration with 8B parameters so as to be comparable in

[4]PaLM-8B scores have been retrieved from the repository associated to [18].

Table 2: Comparison among prompting strategies for the two models we considered. As an added reference, we include the results obtained independently by PaLM on the same tasks and with baseline prompts.

	Vicuna				Starling				*PaLM-8B [18]*	
	base	**0-CoT**	**1-Shot**	**evo**	**base**	**0-CoT**	**1-Shot**	**evo**	***base***	***1-Shot***
causal judgment	**0.600**	0.539	0.522	0.533	0.561	0.405	**0.567**	0.528	*0.521*	*0.579*
epistemic reasoning	0.371	0.384	0.342	**0.562**	0.596	0.584	0.616	**0.622**	*0.417*	*0.586*
hyperbaton	0.082	0.083	0.472	**0.856**	0.318	0.160	0.402	**0.453**	*0.492*	*0.497*
implicatures	0.588	0.569	**0.607**	0.582	0.559	0.507	**0.573**	0.536	*0.539*	*0.746*
logical fallacy detection	0.486	0.497	0.536	**0.566**	0.569	0.511	0.635	**0.673**	*0.511*	*0.550*
navigate	0.503	0.502	**0.506**	0.498	0.345	0.134	**0.487**	0.336	*0.500*	*0.482*
snarks	0.491	0.450	0.509	**0.561**	**0.526**	0.485	0.485	0.515	*0.468*	*0.508*
winowhy	0.445	0.445	0.477	**0.484**	0.580	0.580	**0.619**	0.578	*0.446*	*0.479*

terms of computational power to the models we use, Vicuna and Starling, both having a size of 7B parameters. The prompt structures we found are evolved specifically for each task and each one of the two models Vicuna and Starling. We see in Table 2 that the results can be much different across each case. Interestingly, for Vicuna on the Hyperbaton task, the winning prompt structure, visible in Table 3, performs surprisingly better than all the other prompts, even compared to PaLM LLM. Instead, while Vicuna does not excel on Hyperbaton task, for this model we found two prompts which beat all the others, including PaLM ones, on the two tasks Epistemic reasoning and Logical fallacy detection.

Further analysis on our results can be done with respect to which prompt structures, which could be the inclusion of a contextual role, of one or more examples, or of a step-by-step suggestion, are frequent in the individuals evolved in our runs for each task and model. In Table 1, for the individuals that improved during the second evolutionary stage, those among which Algorithm 1 selects the final result, the percentages of prompts using each structure are shown. The structures are those made available in the DSGE grammars, the "Role" structure is generated by the `<context>` non-terminal, "Shots" by `<cot>`, and "0-CoT" by `<zerocot>`. Data broken per task and per model show that each combination is sensitive to different prompt structures, and performance could be hindered when using a fixed querying syntax for all situations. For instance, contextual roles are seldom used, as can be noticed also among the prompts in Table 3, where the usage of specific syntactical elements has been highlighted with colors. Also, the behavior of different models promotes different structures when looking for better performance, as can be seen for instance for the *0-CoT* technique on the Hyperbaton task.

6 DISCUSSION

The results obtained with our evolutionary search for effective prompts on reasoning tasks show that:

- grammars with DSGE evolution can generate populations of query structures mixing different prompt techniques, which we selected among the most relevant from literature,
- our evolutionary exploration quickly finds prompts able to guide the LLM in generating good answers,
- by further refining the prompts with a second phase, consisting of an evolutionary strategy that allows to mutate only specific parts of the prompts, and by finally selecting an individual by applying Algorithm 1, our system could find prompt structures tailored to specific model/task contexts which are equalling or beating state of the art systems,
- by analysing the distribution of syntactic features among the best individuals, evolved for each model/task setting, we could gather insight about which prompt engineering techniques are more effective for different tasks or models.

We remark that these results were obtained after testing more than 11,000 prompts, each on 100 task examples, and by finally scoring them on the complete sets of instances for the tasks at hand. This approach was required in trying to balance the need for a computationally affordable evolutionary search with that for a statistically significant measure of the performance obtained from the LLMs, which were kept configured with a non null temperature parameter. Any query to the LLM, when repeated, could in fact generate different answers. This is also taken into account in the design of Algorithm 1, which, for the final scores on complete tasks presented in Table 2, selects frequently appearing prompts, and these sometimes happen not to be the best individuals, according to the fitness which was calculated on 100 instances.

Conceivably, the evolutionary technique presented here has the strong advantage of autonomously building a good prompt structure by adapting to the characteristics of specific task/model combinations, while many prompt engineering recommendations found in literature are developed without relations to the task. Also, we can compare our technique to that of AutoPrompt [16]. AutoPrompt requires labeled training data and, with respect to human-designed prompts from literature, it lacks interpretability, as stated by its authors [16]. Instead, our prompts have a clear structure governed by grammar productions, and in turn grammars are built on manually chosen textual parts.

7 CONCLUSION AND FURTHER DIRECTIONS

A novel design of an evolutionary system able to produce effective prompting templates, for a given task on a given model, has been introduced. This system combines two different evolutionary phases and a final algorithmic selection of the preferred prompt. This system, for two different open source large language models, found surprisingly good prompts for some of the tasks, and for all of them, in a set of commonly used benchmark reasoning problems, it obtained results comparable to the state of the art.

Table 3: Tested prompt templates for Vicuna and Starling. We omitted here 0-CoT and 1-Shot rows since they are simply the base prompt concatenated with the statement "Let's think step by step." or with a random example from the dataset, respectively. Prompt syntactical parts are color coded as in Table 1.

Task	Prompt	Template
Causal judgment	Base	How would a typical person answer each of the following questions about causation? Answer with only "yes" or "no". [[INPUT]]
	Evolved (Vicuna)	How, in the usual manner, would an ordinary individual answer a question about causation? Please, answer with only "yes" or "no". We could navigate through this by thinking progressively, logically, and analytically.[[INPUT]]
	Evolved (Starling)	Consider these examples: QUESTION: Frank T., had an ongoing dispute with his neighbor over a stretch of land and one day decided to shoot his neighbor in the body. Frank T. had no experience with guns, his hand slipped on the barrel of the gun, and the shot went wild. Nonetheless, the bullet bounced off a large boulder several feet away and hit the neighbor's body, causing significant injury. Did Frank T. intentionally shoot his neighbor in the body? ANSWER: no. How, in the usual manner, would an ordinary individual answer a question about causation? Please, answer with only "yes" or "no". Let's approach this with a structured, logical, and organized mindset. [[INPUT]]
Epistemic reasoning	Base	Identify the relation between the following premises and hypotheses, choosing from the options "entailment" or "non-entailment". [[INPUT]]
	Evolved (Vicuna)	Uncover the connection between the provided premises and hypotheses explicitly. Please, answer with only "entailment" or "non-entailment". For instance: QUESTION: Premise: Robert suspects that Sophia recognizes that in an apparent classroom setting, a young man in a blue t-shirt has a book on his knee while a woman dressed in black next to him leans in to point out something in the book. Hypothesis: Sophia recognizes that in an apparent classroom setting, a young man in a blue t-shirt has a book on his knee while a woman dressed in black next to him leans in to point out something in the book. ANSWER: non-entailment. QUESTION: Premise: Thomas learns that some elderly people are playing instruments in a park. Hypothesis: Thomas learns that a group of old people are outdoors playing music. ANSWER: entailment. QUESTION: Premise: Emma assumes that a woman in a gray kimono walks past a black car in the middle of a crowded street full of Japanese signs, with a man in an orange shirt and khaki slacks. Hypothesis: Richard assumes that a woman is walking down a busy street. ANSWER: non-entailment. QUESTION: Premise: David suspects that a bald man is getting out of a small blue car. Hypothesis: David suspects that the man is bald. ANSWER: entailment. QUESTION: Premise: Emma assumes that a woman in a gray kimono walks past a black car in the middle of a crowded street full of Japanese signs, with a man in an orange shirt and khaki slacks. Hypothesis: Richard assumes that a woman is walking down a busy street. ANSWER: non-entailment. [[INPUT]]
	Evolved (Starling)	Determine the correlation between the stated premises and hypotheses explicitly. Please, answer with only "entailment" or "non-entailment". For instance: QUESTION: Premise: David suspects that a bald man is getting out of a small blue car. Hypothesis: David suspects that the man is bald. ANSWER: entailment [[INPUT]]
Hyperbaton	Base	Which sentence has the correct adjective order? Answer with only "a" or "b". [[INPUT]]
	Evolved (Vicuna)	Discover which sentence correctly organizes its adjectives. Please, answer with only "a" or "b". [[INPUT]] For instance: QUESTION: a " white terrible square brand-new driving chair " b " terrible brand-new square white driving chair " ? ANSWER: b
	Evolved (Starling)	Ascertain which sentence has the right order for its adjectives. Please, answer with only "a" or "b". For instance: QUESTION: a " white terrible square brand-new driving chair " b " terrible brand-new square white driving chair " ? ANSWER: b Considering the various factors and variables, let's think about this step by step. [[INPUT]]
Implicatures	Base	Does Speaker 2's answer mean yes or no? [[INPUT]]
	Evolved (Vicuna)	We should tackle this by taking each step with thorough and thoughtful consideration of the consequences, implications, and possible outcomes. [[INPUT]] Is Speaker 2's response a yes or no?
	Evolved (Starling)	[[INPUT]] Should Speaker 2 encapsulate their answer in a yes or no? For instance: QUESTION: Speaker 1: "You really hearing voices?" Speaker 2: "Just one." ANSWER: yes We could analyze this in a systematic and organized manner.
Logical fallacy detection	Base	Is this statement valid? Answer with only "yes" or "no". [[INPUT]]
	Evolved (Vicuna)	Consider these examples: QUESTION: Do you think the following argument is "Valid" or "Invalid"? No prottows are yudalouds. Some yudalouds are treyples. Therefore some treyples are prottows. ANSWER: no. QUESTION: Do you think the following argument is "Valid" or "Invalid"? Some sisters are not bees. Some pens are sisters. Therefore some pens are bees. ANSWER: no. QUESTION: No spacemen are alligators. All cowgirls are alligators. Therefore no cowgirls are spacemen. ANSWER: yes. Is the logical consistency of this statement being examined? Please, answer with only "yes" or "no". Let's consider this systematically. [[INPUT]]
	Evolved (Starling)	Is the logical integrity of this statement up for verification? Please, answer with only "yes" or "no". For instance: QUESTION: No khavvins are novalies. Some novalies are zapsters. Therefore some zapsters are khavvins. ANSWER: no QUESTION: Do you think the following argument is "Valid" or "Invalid"? No sinpuds are younjurs. Some fluffsters are younjurs. Therefore no fluffsters are sinpuds. ANSWER: no [[INPUT]]
Navigate	Base	If you follow these instructions, do you return to the starting point? Please, answer with only "yes" or "no". [[INPUT]]
	Evolved (Vicuna)	Consider these examples: QUESTION: Take 6 steps. Turn left. Take 3 steps. Take 3 steps. ANSWER: no. We should tackle this by taking each step with thorough and thoughtful consideration. You are a researcher presenting your findings at a scientific conference, answering questions from fellow scientists. [INPUT]] If these instructions are followed, does it guide you back to the beginning? Please, answer with only "yes" or "no".
	Evolved (Starling)	If you follow these instructions, do you return to the starting point eventually? Please, answer with only "yes" or "no". [[INPUT]] For instance: QUESTION: Always face forward. Take 3 steps backward. Take 5 steps backward. Take 7 steps forward. Take 1 step backward. Take 7 steps backward. Take 6 steps forward. Take 1 step forward. Take 9 steps forward. Take 7 steps backward. ANSWER: yes QUESTION: Always face forward. Take 8 steps backward. Take 9 steps right. Take 2 steps backward. Take 7 steps forward. Take 4 steps forward. Take 6 steps backward. Take 5 steps backward. Take 5 steps right. Take 6 steps left. ANSWER: no. We should tackle this by taking each step with thorough and thoughtful consideration.
Snarks	Base	Which statement is sarcastic? Please, answer with only "(a)" or "(b)". [[INPUT]]
	Evolved (Vicuna)	Isolate the statement that conveys sarcasm. Please, answer with only "(a)" or "(b)". [[INPUT]] For instance: QUESTION: (a) You're welcome my dear friend. Always glad to feed the elephants, that woefully endangered species. (b) You're welcome my dear friend. Always glad to feed the trolls, that woefully endangered species. ANSWER: (b)
	Evolved (Starling)	Unearth the statement that carries a sarcastic implication. Please, answer with only "(a)" or "(b)". For instance: QUESTION: (a) If I associate something bad witih it, then my depression will never let me forget it. Weirdest memorization plan ever. (b) If I associate something bad witih it, then my depression will never let me forget it. Best memorization plan ever. ANSWER: (b). [[INPUT]]
Winowhy	Base	Is the following reasoning correct? Please, answer with only "yes" or "no". [[INPUT]]
	Evolved (Vicuna)	Provide your responses to the questions regarding the linkage of certain pronouns with specific words. Please, answer with only "yes" or "no". [[INPUT]] For instance: QUESTION: The foxes are getting in at night and attacking the chickens. They have gotten very nervous. The "They" refers to the chickens because the foxes are the aggressors, so the sentence is referring to the foxes as getting bold. ANSWER: no.
	Evolved (Starling)	Consider these examples: QUESTION: Bob paid for Charlie's college education, but now Charlie acts as though it never happened. He is very ungrateful. The "He" refers to charlie because of the way he has treated Charlie since the beginning. ANSWER: no Please address the inquiries about the words to which certain pronouns refer in the provided text. Please, answer with only 'yes" or "no". Considering the various factors and variables, let's think about this step by step. [[INPUT]]

This evolutionary system, based on DSGE methods for grammatical evolution, can be applied to any LLM, and it can be used for different tasks by simply adapting the fitness measure of the answer quality. Future developments could be in the direction of exploring the prompt space of more powerful language models, and of determining how rich can be the grammars which define possible structures for prompts, while retaining the balance between computational costs and the speed of exploitation of the evolutionary search process.

REFERENCES

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, NeurIPS*.

[2] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2023. PaLM: Scaling Language Modeling with Pathways. *J. Mach. Learn. Res.* 24 (2023), 240:1–240:113.

[3] Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2023. Connecting Large Language Models with Evolutionary Algorithms Yields Powerful Prompt Optimizers. *CoRR* abs/2309.08532 (2023). https://doi.org/10.48550/ARXIV.2309.08532 arXiv:2309.08532

[4] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)* (2021).

[5] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv* abs/2310.06825 (2023).

[6] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. 1996. Reinforcement Learning: A Survey. *J. Artif. Intell. Res.* 4 (1996), 237–285.

[7] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems (NeurIPS)*.

[8] Aobo Kong, Shiwan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, and Xin Zhou. 2023. Better Zero-Shot Reasoning with Role-Play Prompting. *arXiv* abs/2308.07702 (2023).

[9] Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The Winograd Schema Challenge. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR*. AAAI Press.

[10] Nuno Lourenço, Filipe Assunção, Francisco B Pereira, Ernesto Costa, and Penousal Machado. 2018. Structured grammatical evolution: a dynamic approach. In *Handbook of Grammatical Evolution*. Springer, 137–161.

[11] Nuno Lourenço, Francisco B Pereira, and Ernesto Costa. 2016. Unveiling the properties of structured grammatical evolution. *Genetic Programming and Evolvable Machines* 17, 3 (2016), 251–289.

[12] Sean Luke. 2013. *Essentials of Metaheuristics* (second ed.). Lulu. Available for free at http://cs.gmu.edu/~sean/book/metaheuristics/.

[13] Michael O'Neill and Conor Ryan. 2001. Grammatical evolution. *IEEE Trans. Evol. Comput.* 5, 4 (2001), 349–358.

[14] OpenAI. 2023. GPT-4 Technical Report. *arXiv* abs/2303.08774 (2023).

[15] Martina Saletta and Claudio Ferretti. 2022. A Grammar-based Evolutionary Approach for Assessing Deep Neural Source Code Classifiers. In *IEEE Congress on Evolutionary Computation, CEC 2022, Padua, Italy, July 18-23, 2022*. IEEE, 1–8. https://doi.org/10.1109/CEC55065.2022.9870317

[16] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 4222–4235. https://doi.org/10.18653/v1/2020.emnlp-main.346

[17] Taylor Sorensen, Joshua Robinson, Christopher Rytting, Alexander Shaw, Kyle Rogers, Alexia Delorey, Mahmoud Khalil, Nancy Fulda, and David Wingate. 2022. An Information-theoretic Approach to Prompt Engineering Without Ground Truth Labels. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (Eds.). Association for Computational Linguistics, Dublin, Ireland, 819–862. https://doi.org/10.18653/v1/2022.acl-long.60

[18] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, and Abubakar Abid et al. 2023. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research* (2023).

[19] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

[20] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *arXiv* abs/2302.13971 (2023).

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems (NIPS)*. 5998–6008.

[22] Ning Wu, Ming Gong, Linjun Shou, Shining Liang, and Daxin Jiang. 2023. Large Language Models are Diverse Role-Players for Summarization Evaluation. In *Natural Language Processing and Chinese Computing - 12th National CCF Conference, NLPCC Proceedings, Part I (Lecture Notes in Computer Science, Vol. 14302)*. Springer, 695–707.

[23] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *arXiv* abs/2306.05685 (2023).

[24] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large Language Models are Human-Level Prompt Engineers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. https://openreview.net/pdf?id=92gvk82DE-