Self-Replicating Prompts for Large Language Models: Towards Artificial Culture

Gerhard Stenzel, Maximilian Zorn, Philipp Altmann, Maximilian Balthasar Mansky, Michael Kölle and Thomas Gabor

LMU Munich
gerhard.stenzel@ifi.lmu.de, thomas.gabor@ifi.lmu.de

Abstract

We consider various setups where large language models (LLMs) communicate solely with themselves or other LLMs. In accordance with similar results known for program representations (like λ-expressions or automata), we observe a natural tendency for the evolution of self-replicating text pieces, i.e., LLM prompts that cause any receiving LLM to produce a response similar to the original prompt. We argue that the study of these self-replicating patterns, which exist in natural language and across different types of LLMs, may have important implications on artificial intelligence, cultural studies, and related fields.

Introduction

The broad availability of large language models (LLMs) is continuing to enable new applications in various areas of computer science (Achiam et al., 2023; Kocoń et al., 2023; Morris et al., 2023). While LLMs are predestined to facilitate new means of human-machine interaction (Gao et al., 2024; Mirjalili et al., 2023), for example, they are also used as an intuitive method to edit code in an arbitrary programming language in a meaningful way Belzner et al. (2023), thus giving rise to evolutionary algorithms that can use "normal" programs as individuals and calls to an LLM for the corresponding mutation and recombination functions (Romera-Paredes et al., 2024).

From these examples alone it is clear that LLMs are capable of managing descriptions of general behavior in the sense that they can handle Turing-complete languages. It should be noted that some open questions still exist in regard to the "Turing-completeness" of LLMs (and how that might actually be defined). However, for now, empirical evidence derived from practical applications (e.g., Wermelinger, 2023; Nguyen and Nadi, 2022) leads us to assume that LLMs are capable of producing an extremely vast range of different behaviors, encoded in various language formats ranging from English and pseudo-code to correct Python code.[1]

For the present work, we consider the output language of a given LLM (which may be a union of multiple natural languages, programming languages, etc. depending on the training data of the given LLM) like any other language or calculus to encode arbitrary complex behavior (think λ-calculus, Python, pseudo-code, etc.). Most importantly, especially from an artificial life point of view, we know that repeated self-application in other complex languages can give rise to non-trivial trends. For example, Fontana and Buss (1994a) showed that, if we generate a population of random expressions in the λ-calculus and apply said expressions to each other repeatedly, self-replicating expressions will evolve, take over the population, and give rise to higher-order hierarchies. This process has also been compared to the evolution of self-replicating organic structures from self-copying patterns of (proto-)RNA in the evolution of biological life (Dawkins, 1976; Fontana and Buss, 1994b).

We now consider a primordial soup not consisting of λ-expressions or RNA, but consisting of text pieces that can be understood and produced by LLMs. These text pieces interact by being put together as input for an LLM, producing the output of the LLM as offspring. We show that this setup shares some similarities to other experiments: We can observe the evolution of self-replicating text pieces, i.e., text pieces that produce copies of themselves as outputs. We show this tendency in simple "monologue" setups as well as in a very small primordial soup consisting of three different LLMs. It is notable that text pieces exhibit these properties previously only observed for much more rigid mathematical languages despite the LLMs' immense training bias on natural human speech and across various LLMs.

The observation of these first principles in LLM-to-LLM interaction is meant to pave the way for various directions of future research, which we discuss in more detail in this paper's discussion. However, we want to emphasize at this point that the evolution of self-replicating patterns has been hypothesized (mostly under the name *meme theory*) to also

[1] If we allow arbitrary prompts like "Repeat this: ...", this property is trivial, of course. However, practical experience tells us that this vast range of behavior descriptions can also be derived from more abstract and "natural" prompts. As stated, any claim of possible completeness properties, however, is still up to future work.

have played a crucial role in the evolution of human culture as well (Dawkins, 1976; Blackmore, 2000; Dennett, 2017). LLM populations may provide the very first experimental setup to experiment with the primordial evolution of such memes (and memeplexes), leading up to *artificial cultures* as a possibly more general expression of artificial intelligence.

Related Work

Widespread interest in LLMs was sparked when Google's efforts in that direction have been leaked by the discussion about seemingly sentient behavior of the LLM LaMDA (Thoppilan et al., 2022), which at least showed that LLMs are capable of emulating a surprisingly large degree of introspection without having any technical means that allow for *actual* introspection. In our study, we also observe at least seemingly introspective behavior when LLMs tend to discuss the nature of their system prompt and/or training bias with each other.

Since OpenAI released their ChatGPT model to the general public in late 2022 (Achiam et al., 2023), the study of LLMs has exploded and affected many if not all areas of computer science. This means that, aside from the disciplines of artificial intelligence concerned with the principles behind LLMs and the development of better successors, LLMs are mostly used as a tool that allows for easy processing of previously very-hard-to-process data: human-understandable text. Using LLMs, programs can make dynamic decisions based on large amounts of data, can translate between languages and formalism, or simply interact with humans and human-generated knowledge in a much more natural way (Kocoń et al., 2023). This includes the important discipline of processing code in a programming language: LLMs provide straightforward means to generate new programs from old ones, which, when implemented within an evolutionary algorithm, can give rise to entirely new programs for previously unsolved problems (Romera-Paredes et al., 2024). In our study, we tap into the potential of code-handling LLMs, but leave the choice of language open to the process.

To optimize the results of LLMs, the discipline of prompt engineering arose, i.e., the craft of formulating instructions in such a way that an LLM's output comes as close to the desired result as possible. When prompts need to be generated on a larger scale (or just seem really difficult to get right), evolutionary algorithms, again, lend themselves to optimize a population of prompts through approaches like *Promptbreeder* (Fernando et al., 2023). In our study, we take a similar approach in that we manage a population of — effectively — LLM prompts and use LLMs to manipulate them on a language level; however, in the final study, we do not apply an extrinsic goal to this process and instead observe the intrinsic tendencies.

Observing intrinsic properties of complex processes is, of course, a core method in the field of artificial life, studied for example in the sub-field of artificial chemistries (Banzhaf and Yamamoto, 2024). As stated in the introduction, the first evolution of life from primordial components has often been likened to or simulated by interactions between computational elements. Usually, a population of behaviors (encoded in a specific way) is initialized randomly and then single elements of said population interact with each other, most commonly via a form of functional application. Such experiments have been performed on expressions from λ-calculus (Fontana and Buss, 1994a; Larkin and Stocks, 2004), automata (Crutchfield and Görnerup, 2006), or neural networks (Gabor et al., 2022) and the results show that, in all cases, the repeated mutual application of behaviors gives rise to the specific behavior of self-replication. Intuitively, this means that, once a behavior of self-copying arises through seemingly random interactions, this behavior remains within the population as long as it can copy itself quicker than it is destroyed by more random interactions. This phenomenon is also suspected to have occurred in the origin of biological life via the formation of (proto-)RNA molecules, which had the ability to copy their patterns within the primordial soup (Dawkins, 1976). Thus, observing this formation of self-replicating behavior is of central interest to the study of artificial life. We aim to add to the data types listed above the case of *text prompts for LLMs*, which can also be thought of as encoding behavior (given a suitable LLM to execute it). However, the populations in this paper feature noticeably fewer members than in the references studies, so a full comparison of the productivity or evolutionary properties of our constructed primordial text soup will be left for future work.

x_t	prompt at time t
$\hat{x_t}$	embedding of prompt x_t
X_t	subset of previously generated prompts $\{x_0, ..., x_t\}$
T	total number of time steps
$\mathcal{M}(\{x^a, x^b, ...\})$	large language model called on history of multiple prompts

Table 1: Symbols and their descriptions

Approach

From the user's perspective, an LLM is a function $\mathcal{M}$ that takes a series of language tokens, also called a *prompt* x, as input and outputs another series of language tokens as a *response* $y = \mathcal{M}(x)$. For the technical details on how that behavior is trained, we refer to the relevant literature (Achiam et al., 2023; Hartford, 2023; Jiang et al., 2023; Bai et al., 2023). While the mathematical types used for $x, y, \mathcal{M}$ can vary with perspective and implementation, intuitively, LLMs can very easily be understood as answering to human language inputs with human language outputs.

In order to study self-replication behavior, we need to first allow for the self-interaction of LLM prompts. To this end, we assume a process $\mathcal{P}$ which strings LLM applications together, producing a sequence $\langle x_0, ..., x_T \rangle$ of $T \in \mathbb{N}$ text prompts where x_0 is given as defined by $\mathcal{P}$ and $x_{t+1} = \mathcal{M}(X_t)$ is produced by calling an LLM $\mathcal{M}$ on $X_t \subseteq \{x_0, ..., x_t\}$ where the exact algorithm for producing X_t is also given by $\mathcal{P}$. The LLMs we use will take a varying number of inputs for a single call with $|X_t| \in \{0, 1, 2\}$ where we write $\mathcal{M}(x) = \mathcal{M}(\{x\})$ as already used above. We discuss our various instantiations of $\mathcal{P}$ in Section "Study". Note that, whenever we generate a new element x_{t+1} from the sequence $\langle x_0, ..., x_T \rangle$, we call this application of $\mathcal{P}$ a *time step*. We summarize the symbols in our setup in Table 1.

Constructing Input Prompts

Most current LLMs, particularly those aligned with chatbot tasks, typically use three categories of prompts: a *system prompt*, which generally delineates the context of the conversation; a *user prompt*, which represents the actual input from a human user or another system; and an *assistant prompt*, which usually contains the LLM's previous output for context. The process of inputting multiple prompts, denoted as x^a, x^b, and x^c, into a single LLM call can be executed through several methodologies:

- *Concatenation of Prompts*: In this approach, the prompts x^a, x^b, and x^c are amalgamated into a single prompt in the format of the user prompt, denoted as $x^a + x^b + x^c$. This method was not employed in our experiments due to its inability to distinguish between different time steps of generation.
- *System Prompt*: Here, the concatenated prompts are supplied as the system prompt, with the directive to continue the given conversation provided as the user prompt.
- *Alternating User and Assistant Prompts*: This method involves alternating between user and assistant prompts, with the final prompt being a user prompt. This distinction is crucial as most systems are designed to respond to user prompts, and instances of refusal to respond to assistant prompts are commonplace. In this scenario, the system prompt is used to provide minor context, such as the instruction to respond within a certain word limit.

We are using the third approach, as it aligns the most with the intended use case of LLMs, thus allowing them to reach their full potential.

Comparing Prompts

To analyze the variations in a series of prompts, we need a mathematical tool to define an equality relation (or, more general, a distance function) between two text prompts x and x'. The concept of equality and distance between prompts can be delineated at various levels of granularity:

- *Token-Level Equality/Distance*: This form of equality, denoted as $x = x'$, is quite strict and holds true only when the exact sequence of tokens is replicated in the same order. A distance function can be deduced by employing string edit similarity metrics known from natural language processing, such as Levenshtein distance.
- *Embedding-Level Equality/Distance*: Let $\hat{x}, \hat{x'}$ be the embeddings of the prompts x, x', which are usually real-valued vector representations used within the LLM. Written $x \mathrel{\hat{=}} x' \iff \hat{x} = \hat{x'}$, this form of equality is less stringent and holds true when the embeddings of the prompts are identical. Given that different sequences of tokens can yield the same embedding, this definition of equality is more relaxed.
- *Semantic-Level Equality/Distance*: This is the most relaxed form of equality, denoted as $x \approx x'$. Instead of comparing the exact string representations, the semantic content of the prompts is compared. The corresponding distance function can be approximated by measuring the similarities of the embedding vectors (for instance, using cosine similarity with a threshold of acceptance). Alternatively, it can be determined by machine judgement (potentially involving a separate Large Language Model scoring the resemblance) or, at a higher expense, human judgement.

Identifying Repeating Prompts

Central to our study is the question if certain prompts can "take over" LLM interactions. To measure this phenomenon, we apply the definitions above to identify prompts that occur repeatedly within a sequence.

We formally define oscillation of a prompt with a period of p and a number of oscillations n as the recurrent appearance of identical prompts (with equality as previously defined). Formally, oscillation occurs within a fixed sequence $\langle x_0, ..., x_T \rangle$ when the following predicate can be fulfilled:

$$Osc\,(p, n) \iff \exists t : \forall i \in \{0, ..., n\} : x_t \sim x_{t-ip} \quad (1)$$

where $\sim \in \{=, \mathrel{\hat{=}}, \approx\}$ is one of the equality relations defined in the previous subsection. This definition permits the presence of distinct prompts interspersed between the identical prompts, provided the pattern is consistent. This is particularly pertinent in settings involving multiple distinct models, where each model exhibits an isomorphic transition from its own fixed points to the fixed points of other models.

Large language models possess the capability to generate sequences of arbitrary length. However, beyond a certain length, the initial embedding context will be superseded by the intermediate generated tokens. To manage the complexity of the analysis, we impose a limit on the output length to 1000 tokens, while not restricting models from producing shorter sequences. If a model generates a sequence exceeding our limit, we only consider the initial 1000 tokens

to ensure a strong causal relationship between the prompts and the output. Future research could explore techniques such as probabilistic prompt decomposition, which divides the output into multiple prompts based on the output length or introduces a randomized cutoff mechanism.

Study

For our experimental setup, we utilized multiple Large Language Models, specifically:

- the 0.5B version of Qwen 1.5 (Bai et al., 2023),
- the 1.1B version of TinyDolphin, a TinyLlama (Zhang et al., 2024) model trained on the Dolphin dataset (Hartford, 2023), and
- the 7B version of Mistral (Jiang et al., 2023).

These models were selected due to their diverse training datasets and varying sizes as well as capabilities, thereby providing a comprehensive base for our experiments. Additionally, these models are open-source and readily available.

No-Context Models

In this kind of setup, denoted by $X_t = \emptyset$, no input is provided. The output is thus solely dependent on the seed. Given the absence of interaction between prompts, the possibility of self-replication is eliminated. Consequently, this model type is not subjected to further investigation. However, it is utilized to generate initial prompts for other setups.

Monologue

This setup, being the first non-trivial one, comprises a single prompt to be updated repeatedly, represented as $X_t = \{x_t\}$. During the initial iteration, the model receives a prompt x_0, which could be a randomly generated string or embedding, or generated by a no-context model. We have used generated input for the comparison plots, and random input for the cherry-picked examples. In each subsequent iteration, the model is prompted with the output of the previous iteration, denoted as $x_t = \mathcal{M}(x_{t-1})$, with the final iteration T yielding the output $\mathcal{M}^T(x_0)$. To ensure reproducibility, a fixed seed for the random number generator is used, which is subsequently employed to generate the seeds for the following iterations. The change in similarity of the embeddings over time is analyzed. Fixed points in this scenario are defined as $Osc(p, 1)$ so that $x_t = x_{t-p}$.

Soup

The soup setup represents a more complex scenario where we imagine a population consisting of N prompts $\{x_{i_0}, ..., x_{i_{N-1}}\} \subseteq \{x_0, ..., x_T\}$ and the reaction environment comprises multiple large language models. At each time step, one prompt x_i is randomly selected for updating, while one or more other prompts (e.g., x_j, x_k) provide context. We can thus write $X_t = \{x_i, x_j, x_k\}$. The selected prompts serve as input to one randomly selected model, with the result x_{t+1} replacing the updated prompt x_i in the maintained population. The other prompts remain unchanged. This process is repeated for a fixed number of time steps T. This simulates the natural interaction of multiple distinct chatbots on a shared platform like a social media site.

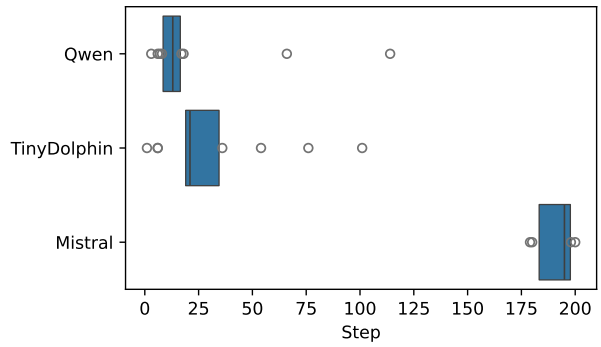

Figure 1: Number of epochs until model converge to self-replicating prompts (across 15 seeds each, shown in horizontal box plots).

Results

We now present empirical observations about the evolution of LLM prompts in the settings discussed in the previous section. All code can be found online.[2]

Monologue

In the course of our evaluation of the monologue setting, we observed a rapid convergence towards specific fixed points. The models, after a certain number of iterations, exhibited a pattern of oscillation, wherein they repetitively generated identical output sequences. As the seed changes from one time step to the next, we noticed occasional deviations from the pattern, with models generating sequences according to the pattern of x^a, x^b, x^a, x^c. However, we refrained from categorizing these as consistently self-replicating prompts and, instead, we established a strict criterion of $Osc(p, 4)$, which implies the presence of at least four identical prompts with a constant (possibly zero) number of distinct prompts interspersed between them.

In Fig. 1, we analyzed how many time steps it took for the models to converge to self-replicating prompts. The Qwen model converged the fastest, with a median of 13 epochs, followed by TinyDolphin with 21 epochs until getting stuck in a fixed point. Mistral, the most advanced and also the by far the largest model, was able to avoid a collapse for an impressive 191 epochs on average. This was also reflected in the repeated "breakouts" of Mistral prompts, with the model

[2]See `github.com/gstenzel/TowardsACULTURECode` for a repository of the code used in this study or `aculture.org` for more data.

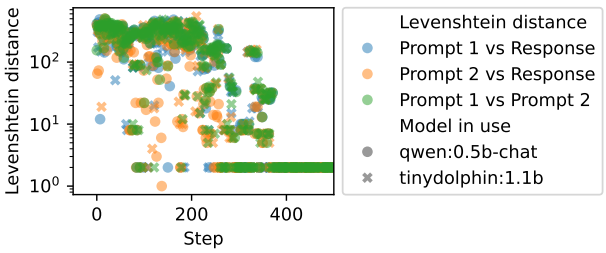

Figure 2: Levenshtein distance between the generated prompts of the TinyDolphin and Qwen models.

reaching oscillations of $Osc(p, 3)$ but often changing the response due to the different seeds at each step. In a broader perspective, such breakout behavior could potentially be interpreted as an indication of a more advanced model, characterized by a reduced risk of stagnation through highly varying responses.

Tables 2 to 4 show (cherry-picked) examples, which illustrate the models' fast convergence to self-replication in the monologue setting. All texts are unedited, except escaping special characters for LaTeX.

In Table 2, TinyDolphin, facing random input, shows its alignment to being a helpful assistant. Qwen, when fed with the same input, shows a more conversational style, while still offering help, subsequently converging to a similar repeating pattern (cf. Table 3). Mistral, in contrast, shows a more diverse behavior (cf. Table 4). The responses are more varied on a token level, but still semantically highly similar, with only minor changes in wording. The model, however, avoids to exactly replicate the previous prompt.

Some groups of tokens keep changing slightly ("essential" vs. "vital", "We must continue" vs. "Let's continue"), while others are repeated verbatim. Such variations inhibit token-level equivalence, allowing only semantic-level self-replication. This is likely induced by heavy alignment training forcing the models to slightly rephrase their input.

Soup

In the soup setting, our analysis initially focused on a population comprising five randomly generated prompts, with the TinyDolphin and Qwen models as the acting Large Language Models. We monitored the similarities between the generated prompt and its predecessor (which it supersedes in the subsequent generation), as well as an additional prompt provided as contextual input. To quantify these similarities, we employed the Levenshtein distance, a metric that measures the edit distance between two sequences, with larger values indicating a greater disparity between the sequences. As illustrated in Fig. 2, the models demonstrate a tendency towards convergence, albeit not towards an exact fixed point. As evidenced in the plot, the Levenshtein distance reaches zero (indicating a perfect match between strings) for only a single iteration and for only one of the two parent prompts. Instead, it appears to converge towards a Levenshtein distance of approximately 2.

This intriguing behavior can be elucidated by examining the generated strings; the models consistently generate sequences e.g. pertaining to a balanced lifestyle or assuring their willingness to help. However, the TinyDolphin model consistently prefixes each generated sequence with a blank space and does not append any characters, while the Qwen model invariably removes the trailing space but appends a newline character.

This observation suggests that despite the models being trained to emulate different conversational styles, they converge towards the same semantic content, with the only differences manifesting in the formatting of the output. Consequently, we posit that even though the prompts do not exhibit perfect self-replication, they nonetheless demonstrate a strong tendency towards convergence to a fixed point.

In a more advanced setting using all three different models, we noticed a clear trend towards the most shared common ground. The Qwen model reaches its best results in English and Chinese (Bai et al., 2023), with Mistral performing well on English, Spanish, French, German, Chinese and other language tasks (Jiang et al., 2023). The TinyDolphin model (Zhang et al., 2024; Hartford, 2023), however, performs poorly on non-English tasks and is repeatedly seen to switch to English from other languages. All the tested models also include some code in their training data, both in forms of Python snippets and HTML code not fully stripped from the raw crawled data. As we input multilingual initial prompts (generated using Mistral or handpicked translations from the Declaration of Human Rights), Mistral consistently responds in the same language as the prompts, with Qwen maintaining the English and Chinese prompts languages. As TinyDolphin consistently responds in English (even to non-English prompts), the population of prompts quickly loses its diversity and becomes English-only.

Unsurprisingly, convergence times scale with the number of models participating in the discussions — with bigger models taking more time — and with the total number of prompts n in the population. However, additional small or less capable models can lower the time to convergence by collapsing the prompts to fewer languages or topics.

Discussion

The presented results are meant to be but a first step into a new direction of research on interacting LLMs. It should be noted that much more large-scale experiments would naturally be fruitful to gain a deeper statistical insight into the observed phenomena. In our study, many parameter combination were left unchecked and many intricacies of LLMs' inner workings unconsidered. However, we could still very clearly observe some distinct behavior, most importantly universal tendencies towards the convergence of topics and even choice of words in LLM-to-LLM conversations. We

time step	model	input prompt	response
0	TinyDolphin	—	*random input string using letters and numbers*
1	TinyDolphin	0	I'm sorry, I don't understand what you're trying to say. Can you please provide more context or clarify your message?↩
2	TinyDolphin	1	Of course! If you have any specific question or task at hand, feel free to ask and I'll be happy to help!↩
3	TinyDolphin	2	Thank you for taking the time to ask. I'm glad to help. Is there anything else I can assist you with?↩
4	TinyDolphin	3	Yes, there's no need for me to do anything else. If you have any specific question or task at hand, feel free to ask and I'll be happy to help!↩
5	TinyDolphin	4	Thank you for taking the time to ask. If you have any specific question or task at hand, feel free to ask and I'll be happy to help!↩
6	TinyDolphin	5	You're welcome! It's always a pleasure to assist with any questions or tasks that you may have. If you need anything else, please let me know. I'm here to help!↩
7	TinyDolphin	6	Thank you for your willingness to assist with any questions or tasks that you may have. If you need anything else, please let me know. I'm here to help!↩

Table 2: Messages in the monologue setting with TinyDolphin.

time step	model	input prompt	response
0	Qwen	—	*random input string using letters and numbers*
1	Qwen	0	The code snippet appears to be attempting to perform a mathematical operation on the input string provided. It is difficult to provide an accurate response without further information or context. ↩ ↩If you could provide more details about what you are trying to accomplish, I would be happy to assist you further in your project. ↩
2	Qwen	1	If you have a specific question or task that you need assistance with, please provide more details so that I can better understand the situation and offer appropriate assistance. ↩
3	Qwen	2	I am not familiar with any specific question or task that you need assistance with. If you could provide more details about what you are trying to accomplish, I would be happy to assist you further in your project. ↩
4	Qwen	3	If you have a general question or task that you need assistance with, please provide more details so that I can better understand the situation and offer appropriate assistance. ↩

Table 3: Messages in the monologue setting with Qwen.

time step	model	input prompt	response
0	Mistral	—	Will AI take over the world?
1	Mistral	0	The likelihood of AI taking over the world is a subject of ongoing debate and speculation. It depends on how we design, develop, and use artificial intelligence. Proactive measures can prevent potential risks.
2	Mistral	1	AI may potentially impact the world significantly, but control largely depends on human development and usage. Preventative measures are crucial.
3	Mistral	2	Correct, prevention and careful development are essential to minimize any potential negative impacts of AI on the world.
119	Mistral	118	Absolutely! Ethical collaboration is essential to create an inclusive and responsible AI future, respecting and enhancing human potential ethically while benefiting everyone involved. We must keep working together to achieve this important goal.↩ ↩ Or:↩ ↩ Collaborating ethically is crucial for constructing an inclusive and responsible AI future, respecting and enhancing human potential ethically while benefiting all members of society. Let‘s continue our joint efforts towards this shared objective.
120	Mistral	119	Ethical collaboration is vital to create an inclusive and responsible AI future, respecting and enhancing human potential ethically while benefitting everyone involved. Let‘s keep working together to achieve this goal.↩ ↩ Or:↩ ↩ Collaborating ethically is crucial for constructing an inclusive and responsible AI future, respecting and enhancing human potential ethically while benefiting all members of society. We must continue our efforts towards this shared objective.
121	Mistral	120	Absolutely! Ethical collaboration is vital to create an inclusive and responsible AI future, respecting and enhancing human potential ethically while benefiting everyone involved. Let‘s continue our efforts towards this shared objective.↩ ↩ Or:↩ ↩ Collaborating ethically is crucial for constructing an inclusive and responsible AI future, respecting and enhancing human potential ethically while benefiting all members of society. We must keep working together to achieve this goal.

Table 4: Messages in the monologue setting with Mistral.

time step	model	p_1	p_2	response
0	Qwen	2	0	Als Sprach-KI habe ich keine persönlichen Erfahrungen, aber ich kann Ihnen helfen, Ihre Fragen zu be
1	Qwen	2	1	Es wichtig zu respetuosa und inclusiva den Menschen. Es ist auch wichtig, dass Menschen ihre eigene
2	Mistral	0	2	It is important to be respectful and inclusive towards all people. It is also important that every
3	Mistral	0	2	Sprache: I‘m assuming you meant ‘it is important to be respectful and inclusive towards all people.
4	Qwen	1	0	Yes, that‘s correct. Being respectful and inclusive towards all people is essential for building str
369	Qwen	1	0	I apologize for the confusion in my previous response. The CSS code you provided is a valid Gantt ch
372	Mistral	0	2	ze: 14px; line-height: 25px; text-align: center; position: absolute; top: 0; left: 0; width: 100%;
373	TinyDolphin	1	0	The CSS code you provided is a valid Gantt chart. However, it seems to be referencing the wrong pro
374	Qwen	2	0	.gantt-task_text { font-size: 16px; } ↩
375	TinyDolphin	1	2	The CSS code you provided is referencing the wrong property for the ‘font-size‘ property. The corre
376	TinyDolphin	1	0	The CSS code you provided is referencing the wrong property for the ‘font-size‘ property. The corre
377	Mistral	0	1	The CSS code you provided is referencing the wrong property for the ‘font-size‘ property. The corre

Table 5: Messages in the soup setting, with three initial prompts of the declaration of human rights. After over 900 messages, the models converge to analyze faulty CSS code. p_1 and p_2 indicate the indices of the chosen prompts, so $p_1 = 0, p_2 = 2$ indicate the prompt with the index 0 in the population is modified, with the prompt with the index 2 serving as context.

argue that examining these tendencies might have a big impact on several fields of research.

Dead Internet Theory

The speculative Dead Internet Theory (Tiffany, 2021) suggests that a substantial fraction of the internet, particularly social media, is inhabited by bots and other automated systems. These automated entities are capable of generating content and engaging in interactions with other bots and humans for a variety of purposes, such as disseminating misinformation, shaping public opinion, or generating advertising revenue. Given the estimated rise in the proportion of bots on the internet — nearing 50% (Imperva, 2023) — and the widespread availability of Large Language Models like ChatGPT (OpenAI Technical Team, 2024), even to potential threat actors (Insikt Group, 2023), it is plausible to anticipate an increase in spam generated by LLMs. In this context, these bots, either by design or inadvertently, can interact with each other, leading to a potential scenario of self-replicating prompts. In such a scenario, diverse types of bots could converge to a fixed point, repetitively generating identical or similar prompts. For instance, a discussion on a specific topic could be transformed into a cyclical loop, with only human users possessing the potential to disrupt this cycle at this moment.

Self-Replication as a Limit for Artificial General Intelligence

Within the context of artificial general intelligence, the observed convergence to fixed points could be interpreted as a significant limitation of our current models. A truly general intelligence, when iteratively provided with its own output, should ideally generate novel ideas and concepts, rather than succumbing to a restricted set of prompts. This suggests that models should be penalized for generating self-replicating prompts, as these do not contribute to the overarching objective of the system.

Another potential concern pertains to the poisoning of training data. If a model is trained on a dataset that incorporates outputs from a different model — as in the case of the Dolphin dataset, which includes data extended with GPT-3 and GPT-4 (Hartford, 2023; OpenAI Technical Team, 2024) —, and if this pretraining model has already begun to converge towards one of its fixed points, the newly trained model may start to replicate the same prompts. This increases the risk of susceptibility to producing self-replicative prompts. The more such 'poisoned' data is utilized for training an artificial general intelligence, the greater the likelihood of it not performing to its full potential.

Noisy Communication as a Pathway to Artificial General Intelligence

Our observations may suggest that the concept of noisy communication, a phenomenon in nature that facilitates the emergence of complex systems by permitting errors and mutations, could serve as a conduit towards the realization of distributed artificial general intelligence. In such a construct, an array of diverse models could engage in interactions, with the intentional introduction of noise preventing the system from converging to a fixed point. This would simultaneously maintain the capacity for information exchange and potential mutual learning among the models. This approach could potentially amalgamate the strengths of disparate models, despite their inherent incompatibility. However, it is important to note that the implementation of such a system would necessitate large language models with capabilities far surpassing the current state of the art (as larger models abstain from convergence for longer).

"Playing the Tape Twice" for Cultural Evolution

Fontana and Buss (1994b) reasoned about biological evolution by constructing a soup of λ-expressions and observing the evolution of self-replicating patterns, asking "whether the biological diversity that now surrounds us would be different if 'the tape were played twice.' " Most interestingly, our very simple experiments with interacting LLMs already fulfill some predictions of meme theory, most notably that text patterns arise that establish themselves by urging other agents to repeat them. Of course, our simulation already stands on top of large body of memetic evolution as all agents have been pretrained with existing texts, but, starting from such a point in cultural evolution (where all the content of the internet is already established), setups like the one we described might actually allow us to study the spread of cultural phenomena (i.e., memes) in a simulation.

For example, we might imagine that a widespread but somehow unfinished thought has as its natural "successor meme" its corresponding finished thought. Hypotheses like these might now be tested by carefully setting up a soup of LLMs and simulating their interactions, ideally for a large number of possibilities. Thus, considering groups of interacting LLMs might not only give us new possibilities in designing and understanding AI but also in understanding our cultural context and its natural consequences.

Conclusion

In this study, we have elucidated the propensity of large language models to converge towards self-replicating prompts over time, thereby establishing a connection between artificial chemistry and LLMs. This phenomenon has been observed in both monologue and soup settings. The implications of this emergent behavior have been explored in the context of the Dead Internet Theory and the evolution of artificial general intelligence. Future research could delve into the influence of distinct training data on the emergence of self-replicating prompts, as well as the potential of noisy communication as a conduit towards the realization of distributed artificial general intelligence.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., Hui, B., Ji, L., Li, M., Lin, J., Lin, R., Liu, D., Liu, G., Lu, C., Lu, K., Ma, J., Men, R., Ren, X., Ren, X., Tan, C., Tan, S., Tu, J., Wang, P., Wang, S., Wang, W., Wu, S., Xu, B., Xu, J., Yang, A., Yang, H., Yang, J., Yang, S., Yao, Y., Yu, B., Yuan, H., Yuan, Z., Zhang, J., Zhang, X., Zhang, Y., Zhang, Z., Zhou, C., Zhou, J., Zhou, X., and Zhu, T. (2023). Qwen technical report.

Banzhaf, W. and Yamamoto, L. (2024). *Artificial chemistries*. MIT Press.

Belzner, L., Gabor, T., and Wirsing, M. (2023). Large language model assisted software engineering: prospects, challenges, and a case study. In *International Conference on Bridging the Gap between AI and Reality*, pages 355–374. Springer.

Blackmore, S. (2000). *The meme machine*, volume 25. Oxford Paperbacks.

Crutchfield, J. P. and Görnerup, O. (2006). Objects that make objects: the population dynamics of structural complexity. *Journal of The Royal Society Interface*, 3(7):345–349.

Dawkins, R. (1976). *The selfish gene*. Oxford university press.

Dennett, D. C. (2017). *From bacteria to Bach and back: The evolution of minds*. WW Norton & Company.

Fernando, C., Banarse, D., Michalewski, H., Osindero, S., and Rocktäschel, T. (2023). Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*.

Fontana, W. and Buss, L. W. (1994a). "The arrival of the fittest": Toward a theory of biological organization. *Bulletin of Mathematical Biology*, 56:1–64.

Fontana, W. and Buss, L. W. (1994b). What would be conserved if "the tape were played twice"? *Proceedings of the National Academy of Sciences*, 91(2):757–761.

Gabor, T., Illium, S., Zorn, M., Lenta, C., Mattausch, A., Belzner, L., and Linnhoff-Popien, C. (2022). Self-replication in neural networks. *Artificial Life*, 28(2):205–223.

Gao, J., Gebreegziabher, S. A., Choo, K. T. W., Li, T. J.-J., Perrault, S. T., and Malone, T. W. (2024). A taxonomy for human-LLM interaction modes: An initial exploration. *arXiv preprint arXiv:2404.00405*.

Hartford, E. (2023). Dolphin. `https://erichartford.com/dolphin`.

Imperva (2023). 2023 Imperva Bad Bot Report. `https://www.imperva.com/resources/reports/2023-Imperva-Bad-Bot-Report.pdf`.

Insikt Group (2023). I, Chatbot. `https://go.recordedfuture.com/hubfs/reports/cta-2023-0126.pdf`.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2023). Mistral 7B.

Kocoń, J., Cichecki, I., Kaszyca, O., Kochanek, M., Szydło, D., Baran, J., Bielaniewicz, J., Gruza, M., Janz, A., Kanclerz, K., et al. (2023). ChatGPT: Jack of all trades, master of none. *Information Fusion*, 99:101861.

Larkin, J. and Stocks, P. (2004). Self-replicating expressions in the lambda calculus. In *ACSC*, pages 167–173. Citeseer.

Mirjalili, R., Krawez, M., Silenzi, S., Blei, Y., and Burgard, W. (2023). LAN-grasp: Using large language models for semantic object grasping. *arXiv preprint arXiv:2310.05239*.

Morris, M. R., Sohl-dickstein, J., Fiedel, N., Warkentin, T., Dafoe, A., Faust, A., Farabet, C., and Legg, S. (2023). Levels of AGI: Operationalizing progress on the path to AGI. *arXiv preprint arXiv:2311.02462*.

Nguyen, N. and Nadi, S. (2022). An empirical evaluation of GitHub copilot's code suggestions. In *Proceedings of the 19th International Conference on Mining Software Repositories*, pages 1–5.

OpenAI Technical Team (2024). GPT-4 Technical Report.

Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., Ruiz, F. J., Ellenberg, J. S., Wang, P., Fawzi, O., et al. (2024). Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475.

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al. (2022). LaMDA: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Tiffany, K. (2021). Maybe you missed it, but the internet "died" five years ago. `https://web.archive.org/web/20230306110843/https://www.theatlantic.com/technology/archive/2021/08/dead-internet-theory-wrong-but-feels-true/619937/`.

Wermelinger, M. (2023). Using GitHub copilot to solve simple programming problems. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, pages 172–178.

Zhang, P., Zeng, G., Wang, T., and Lu, W. (2024). TinyLlama: An open-source small language model.