Bioinformatics, 2024, **40(2)**, btae075
https://doi.org/10.1093/bioinformatics/btae075
Advance Access Publication Date: 10 February 2024
Original Paper

OXFORD

Data and text mining

GeneGPT: augmenting large language models with domain tools for improved access to biomedical information

Qiao Jin [1], **Yifan Yang**[1], **Qingyu Chen** [1], **Zhiyong Lu** [1,*]

[1]National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20894, United States

*Corresponding author. National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20894, United States. Tel: 301-594-7089, E-mail: zhiyong.lu@nih.gov (Z.L.)

Associate Editor: Jonathan Wren

Abstract

Motivation: While large language models (LLMs) have been successfully applied to various tasks, they still face challenges with hallucinations. Augmenting LLMs with domain-specific tools such as database utilities can facilitate easier and more precise access to specialized knowledge. In this article, we present GeneGPT, a novel method for teaching LLMs to use the Web APIs of the National Center for Biotechnology Information (NCBI) for answering genomics questions. Specifically, we prompt Codex to solve the GeneTuring tests with NCBI Web APIs by in-context learning and an augmented decoding algorithm that can detect and execute API calls.

Results: Experimental results show that GeneGPT achieves state-of-the-art performance on eight tasks in the GeneTuring benchmark with an average score of 0.83, largely surpassing retrieval-augmented LLMs such as the new Bing (0.44), biomedical LLMs such as BioMedLM (0.08) and BioGPT (0.04), as well as GPT-3 (0.16) and ChatGPT (0.12). Our further analyses suggest that: First, API demonstrations have good cross-task generalizability and are more useful than documentations for in-context learning; second, GeneGPT can generalize to longer chains of API calls and answer multi-hop questions in GeneHop, a novel dataset introduced in this work; finally, different types of errors are enriched in different tasks, providing valuable insights for future improvements.

Availability and implementation: The GeneGPT code and data are publicly available at https://github.com/ncbi/GeneGPT.

1 Introduction

Large language models (LLMs) such as PaLM (Chowdhery *et al.* 2022) and GPT-4 (OpenAI 2023) have shown great success on a wide range of general-domain Natural Language Processing (NLP) tasks. They also achieve state-of-the-art (SOTA) performance on domain-specific tasks like clinical trial matching (Jin *et al.* 2023b, Yuan *et al.* 2023, Wong *et al.* 2023) and biomedical question answering (Singhal *et al.* 2022, Liévin *et al.* 2022, Nori *et al.* 2023, Tian *et al.* 2024). However, since there is no intrinsic mechanism for auto-regressive LLMs to "consult" with any source of truth, they can generate plausible-sounding but incorrect content (Ji *et al.* 2023). To tackle the hallucination issue, various studies have been proposed to augment LLMs (Mialon *et al.* 2023) by either conditioning them on retrieved relevant content (retrieval-augmented generation) (Guu *et al.* 2020, Lewis *et al.* 2020, Borgeaud *et al.* 2022, Jin *et al.* 2023a) or allowing them to use other external tools such as program APIs (tool augmentation) (Gao *et al.* 2022, Parisi *et al.* 2022, Yao *et al.* 2022, Schick *et al.* 2023, Qin *et al.* 2023).

In this work, we propose to teach LLMs to use the Web APIs of the National Center for Biotechnology Information (NCBI) by directly generating the request URLs. NCBI provides API access to its entire biomedical databases and tools, including Entrez Programming Utilities (E-utils) and BLAST URL API (Altschul *et al.* 1990, Schuler *et al.* 1996, Sayers *et al.* 2019). Enabling LLMs to use NCBI Web APIs can provide easier and more precise access to biomedical information, especially for users who are inexperienced with the database systems. More importantly, Web APIs can relieve users from locally implementing functionalities, maintaining large databases, and heavy computation burdens because the only requirement for using Web APIs is an Internet connection.

We introduce GeneGPT, a novel method that prompts Codex (Chen *et al.* 2021) to use NCBI Web APIs by in-context learning (Brown *et al.* 2020). GeneGPT consists of two main modules: (a) a specifically designed prompt that consists of documentations and demonstrations of API usage, and (b) an inference algorithm that integrates API calls in the Codex decoding process. We evaluate GeneGPT on GeneTuring (Hou and Ji 2023), a question answering (QA) benchmark for genomics, and compare GeneGPT to a variety of other LLMs such as the new Bing (https://www.bing.com/new), ChatGPT (https://chat.openai.com/), and BioGPT (Luo *et al.* 2022). GeneGPT achieves the best performance on eight GeneTuring tasks with an average score of 0.83, which is remarkably higher than the previous SOTA (0.44 by New Bing). In addition, we systematically characterize GeneGPT and find that: First, API demonstrations are more useful than documentations for in-context learning; second, GeneGPT generalizes to longer chains of subquestion decomposition and API calls with simple demonstrations; finally, GeneGPT makes specific errors that are enriched for each task.

Received: 13 November 2023; **Revised:** 8 January 2024; **Editorial Decision:** 31 January 2024; **Accepted:** 8 February 2024

In summary, our contributions are 3-fold:

1) We introduce GeneGPT, a novel method that uses NCBI Web APIs to answer biomedical questions. To the best of our knowledge, this is one of the first studies on augmenting LLMs with domain-specific Web API tools.
2) GeneGPT achieves SOTA performance on 8 tasks in the GeneTuring benchmark, largely outperforming previous best results by 88% (0.83 versus 0.44 set by the new Bing).
3) We conduct experiments to further characterize GeneGPT, including ablation, probing, and error analyses. We also contribute a novel GeneHop dataset, and use it to show that GeneGPT can perform chain-of-thought API calls to answer multi-hop genomics questions.

2 GeneGPT

In this section, we first introduce the general functions and syntax of NCBI Web APIs. We then describe two key components of GeneGPT: its prompt design for in-context learning and the inference algorithm.

2.1 NCBI Web APIs

We utilize NCBI Web APIs of E-utils (https://www.ncbi.nlm.nih.gov/books/NBK25501/) that provide access to biomedical databases and the BLAST tool (https://ncbi.github.io/blast-cloud/dev/api.html) for DNA sequence alignment. Web API calls are implemented by the `urllib` library in Python.

2.1.1 E-utils

It is the API for accessing the Entrez portal (Schuler *et al.* 1996), which is a system that covers 38 NCBI databases of biomedical data like genes and proteins (Sayers *et al.* 2019). The E-utils API provides a fixed URL syntax for rapidly retrieving such information. Specifically, the base URL for an E-utils request is "https://eutils.ncbi.nlm.nih.gov/entrez/eutils/function.fcgi," where `function` can be `esearch`, `efetch`, or `esummary`. Typically, the user first calls `esearch` to get the unique database identifiers of a given query term. Then, `efetch` or `esummary` can be called to get the full records or text summaries of a given list of identifiers returned by `esearch`. Important arguments in the URL request include the search term or ids (`term` or `id`), the database to use (`db`), the maximum number of returned items (`retmax`), and the return format (`retmode`).

2.1.2 BLAST URL API

BLAST takes as input a sequence of nucleotides or amino acids and finds the most similar sequences in the database (Altschul *et al.* 1990, Boratyn *et al.* 2013). The results can be used to infer relationships between sequences or identify members of gene families. The BLAST API allows users to find regions of similarities between nucleotide or protein sequences to existing databases using the BLAST algorithm. The base URL for the API is "https://blast.ncbi.nlm.nih.gov/blast/Blast.cgi." By sending different parameters to this API, the user can submit and retrieve queries that are executed by NCBI servers. Every call to the API must include a `CMD` parameter that defines the type of the call. When submitting queries using `CMD=Put`, the user can specify the querying database with the `DATABASE` parameter, the searching program with the `PROGRAM` parameter, and the query sequence with the `QUERY` parameter. The user will get an `RID` after the `CMD=Put` API call, and can make another API call with the `Get` command and the returned `RID` to retrieve its BLAST results.

2.2 In-context learning

We teach an LLM to use NCBI Web APIs through in-context learning with a carefully designed prompt. Figure 1 shows an example of the GeneGPT prompt, which is composed of four modules: (i) an instruction; (ii) API documentations; (iii) API demonstrations; and (iv) a test question. The first three parts are fixed for all tasks, while the last one is task-specific.

1) **Instruction:** The prompt starts with an overall task description ("Your task is to use NCBI APIs to answer genomic questions."). It is then followed by documentations and demonstrations of API usage, which are summarized in Table 1.
2) **Documentations (Dc.)** provide natural language descriptions of the API functionality, general syntax, and argument choices. We include one for the E-utils API (Dc.1) and one for the BLAST tool (Dc.2).
3) **Demonstrations (Dm.)** are concrete examples of using NCBI Web APIs to solve questions. Based on questions in the GeneTuring tasks, we manually write four demonstrations that cover four functions (`esearch`, `efetch`, `esummary`, `blastn`) and four databases (`gene`, `snp`, `omim`, `nt`) of E-utils and BLAST. The API URLs and the call results are marked up by "[]," with a special "->" symbol inserted in between that serves as an indicator for API calls.
4) **Test question:** The specific test question is then appended to the end of the prompt.

While the initial GeneGPT uses all documentations and demonstrations (denoted as GeneGPT-full in Table 2), we find through analyses in §4.1 that GeneGPT can work well with only two demonstrations (GeneGPT-slim) on all tasks.

2.3 Inference algorithm

The GeneGPT inference algorithm is briefly shown in Fig. 2. Specifically, we first append the given question to the prompt (described in §2.2) and feed the concatenated text to Codex (`code-davinci-002`, (Chen *et al.* 2021)) with a temperature of 0. We choose to use Codex for two reasons: (i) it is pre-trained with code data and shows better code understanding abilities, which is crucial in generating the URLs and interpreting the raw API results; (ii) its API provides a reasonably long context length (8k tokens) among all available models at the time of writing so that we can fit the demonstrations in. We also experiment with two more recent implementation and prompting choices: (i) we replace the Codex with gpt-3.5-turbo-16k-0301 for the base LLM: (ii) we provide the same set of NCBI Web API tools for the ReAct agent (Yao *et al.* 2022) implemented in LangChain (https://www.langchain.com/).

We discontinue the text generation when the "->" symbol is detected, which is the indication for an API call request. Then we extract the last URL and call the NCBI Web API with the request. The raw execution results will be appended to the generated text, and it will be fed to Codex to continue the generation. When "\n\n," an answer indicator used in

Figure 1. (Left) GeneGPT uses NCBI Web API documentations and demonstrations in the prompt for in-context learning. (Right) Examples of GeneGPT answering GeneTuring and GeneHop questions with NCBI Web APIs

Table 1. Summary of API usage documentations (Dc.1 and Dc.2) and demonstrations (Dm.1–4) in the GeneGPT prompt. Complete texts are shown in Supplementary Appendix A.

Comp.	Documentation	Database	Function
Dc.1	E-utils	`gene`, `snp`, `omim`	`esearch`, `efetch`, `esummary`
Dc.2	BLAST	`nt`	`blastn`
Dm.1	Alias	`gene`	`esearch->efetch`
Dm.2	Gene SNP	`snp`	`esummary`
Dm.3	Gene disease	`omim`	`esearch->esummary`
Dm.4	Alignment	`nt`	`blastn`

the demonstrations, is generated, we will stop the inference and extract the answer after the generated "Answer:" span.

3 Experiments

3.1 GeneTuring

The GeneTuring benchmark (Hou and Ji 2023) contains 12 tasks, and each task has 50 question-answer pairs. We use 9 GeneTuring tasks that are related to NCBI resources to evaluate the proposed GeneGPT model, and the QA samples are shown in Supplementary Appendix B. The chosen tasks are classified into four modules and briefly described in this section.

3.1.1 Nomenclature

The nomenclature tasks include the gene alias task and the gene name conversion task, where the objective is to find the official gene symbols for their non-official synonyms.

3.1.2 Genomics location

The tasks are about the locations of genes, SNP, and their relations. We include the gene location, SNP location, and gene SNP association tasks. The first two tasks ask for the chromosome locations (e.g. "chr2") of a gene or an SNP, and the last one asks for related genes for a given SNP.

3.1.3 Functional analysis

It is related to gene functions. We use the gene disease association task, where the goal is to return related genes for a given disease, and the protein-coding genes task, which asks whether a gene codes a protein.

3.1.4 Sequence alignment

The tasks query specific DNA sequences. We use the DNA sequence alignment to human genome task and the DNA sequence alignment to multiple species task. The former maps a DNA sequence to a specific human chromosome (e.g. "chr13"), while the latter maps an DNA sequence to a specific species (e.g. "zebrafish").

3.2 Compared methods

We evaluate four settings of GeneGPT, a full setting (GeneGPT-full) where all prompt components are used, as well as a slim setting (GeneGPT-slim) inspired by our ablation and probing analyses (§4.1) where only Dm.1 and Dm.4 are used. We also replace Codex with gpt-3.5-turbo-16k as the base LLM (GeneGPT-turbo), and implement the NCBI tool set with LangChain's ReAct agent (GeneGPT-lang).

We compare GeneGPT with various baselines listed in Hou and Ji (2023), including general-domain GPT-based (Radford *et al.* 2018) LLMs such as GPT-2 (Radford *et al.* 2019), GPT-3 (`text-davinci-003`) (Brown *et al.* 2020), and ChatGPT (https://chat.openai.com/; Jan 31 version), GPT-2-sized biomedical domain-specific LLMs such as BioGPT (Luo *et al.* 2022)

Table 2. Performance of GeneGPT compared to other LLMs on GeneTuring.

GeneTuring task	GPT-2	BioGPT	BioMedLM	GPT-3	ChatGPT	New Bing	GeneGPT (ours)			
							-full	-slim	-turbo	-lang
Nomenclature										
Gene alias	0.00	0.00	0.04	0.09	0.07	0.66	0.80*	**0.84***	0.64*	0.76*
Gene name conversion	0.00	0.00	0.00	0.00	0.00	0.85	**1.00**	**1.00**	**1.00**	0.02
Average	0.00	0.00	0.02	0.05	0.04	0.76	0.90	**0.92**	0.82	0.39
Genomic location										
Gene SNP association	0.00	0.00	0.00	0.00	0.00	0.00	**1.00***	**1.00**	0.96	0.90
Gene location	0.01	0.04	0.12	0.09	0.09	0.61	0.62	**0.66**	0.54	0.54
SNP location	0.03	0.05	0.01	0.02	0.05	0.01	**1.00**	0.98	0.98	0.74
Average	0.01	0.03	0.04	0.04	0.05	0.21	0.87	**0.88**	0.82	0.73
Functional analysis										
Gene disease association	0.00	0.02	0.16	0.34	0.31	**0.84**	0.76*	0.66	0.63	0.39
Protein-coding genes	0.00	0.18	0.37	0.70	0.54	0.97	0.76	**1.00**	0.96	0.90
Average	0.00	0.10	0.27	0.52	0.43	**0.91**	0.76	0.84	0.80	0.65
Sequence alignment										
DNA to human genome	0.02	0.07	0.03	0.00	0.00	0.00	**0.44***	**0.44***	0.42*	0.06*
DNA to multiple species	0.02	0.00	0.00	0.20	0.00	0.00	0.86	**0.88**	**0.88**	0.54
Average	0.02	0.04	0.02	0.10	0.00	0.00	0.65	**0.66**	0.65	0.30
Overall average	0.00	0.04	0.08	0.16	0.12	0.44	0.80	**0.83**	0.78	0.54

* One-shot learning for GeneGPT.
Bolded and underlined denote the highest and second-highest results, respectively. -full and -slim are the main GeneGPT implementations proposed in this work. -turbo: GeneGPT with gpt-3.5-turbo-16k. -lang: GeneGPT implemented with the LangChain ReAct agent.

Input: question
Model: Codex (`code-davinci-002`)
Output: answer
prompt ← header + demonstrations + question
finished ← False
while not finished **do**
  next token ← Codex(prompt)
  prompt ← prompt + next token
  if next token is "->" **then**
    url ← extractLastURL(prompt)
    result ← callWebAPI(url)
    prompt ← prompt + result
  else if next token is "\n\n" **then**
    answer ← extractAnswer(prompt)
    finished ← True
  end if
end while

Figure 2. GeneGPT inference algorithm

and BioMedLM (https://crfm.stanford.edu/2022/12/15/biomedlm.html) (PubMedGPT), as well as the new Bing (https://www.bing.com/new), a retrieval-augmented LLM that has access to relevant web pages retrieved by Bing.

3.3 Evaluation

For the performance of the compared methods, we directly use the results reported in the original benchmark that are manually evaluated.

To evaluate our proposed GeneGPT method, we follow the general criteria but perform automatic evaluations. Specifically, we only consider *exact* matches between model predictions and the ground truth as correct predictions for all nomenclature and genomics location tasks. For the gene disease association task, we measure the recall as in the original dataset but based on *exact* individual gene matches. For the protein-coding genes task and the DNA sequence alignment to multiple species task, we also consider *exact* matches as correct after applying a simple vocabulary mapping that converts model-predicted "yes"/"no" to "TRUE"/"NA" and Latin species names to their informal names (e.g. "*Saccharomyces cerevisiae*" to "yeast"), respectively. For the DNA sequence alignment to human genome task, we give correct chromosome mapping but incorrect position mapping a score of 0.5 (e.g. chr8:7081648–7081782 versus chr8:1207812–1207946), since the original task does not specify a reference genome. Overall, our evaluation of GeneGPT is more strict than the original evaluation of other LLMs in Hou and Ji (2023), which performs manual evaluation and might consider non-exact matches as correct.

3.4 Main results

Table 2 shows the performance of GeneGPT on the GeneTuring tasks in comparison with other LLMs. For GeneGPT, tasks with "*" in Table 2 are one-shot where one instance is used as API demonstration, and the other tasks are zero-shot. For the compared LLMs, all tasks are zero-shot.

3.4.1 Nomenclature

GeneGPT achieves SOTA performance on both the one-shot gene alias task with an accuracy of 0.84 and the zero-shot gene name conversion task with an accuracy of 1.00. On average, GeneGPT outperforms New Bing by a large margin (0.92 versus 0.76). All other GPT models have accuracy scores of less than 0.10 on the nomenclature tasks.

3.4.2 Genomic location

GeneGPT also achieves SOTA performance on all genomic location tasks, including the gene SNP association task (1.00) gene location task (0.66), and the SNP location task (1.00). While the New Bing is comparable to GeneGPT on gene location (0.61 versus 0.66), its performance on the two SNP-related tasks is close to 0. Again, most other LLMs score less than 0.10. Notably, while all genomics location tasks are zero-shot for GeneGPT-slim, it performs comparably to GeneGPT-full which uses one gene SNP association

demonstration. This indicates that API demonstrations have strong cross-task generalizability.

3.4.3 Functional analysis

The new Bing performs better functional analysis tasks than GeneGPT (average score: 0.91 verusu 0.84), which is probably because many web pages related to gene functions can be retrieved by Bing search. We also note that other LLMs, especially GPT-3 and ChatGPT, perform moderately well and much better than they perform on other tasks, perhaps due to the fact that many gene-function-related texts are included in their pre-training corpora.

3.4.4 Sequence alignment

GeneGPT performs much better with an average score of 0.66 than all other models including New Bing, which essentially fails on the sequence alignment tasks. This is not surprising since sequence alignment is easy with the BLAST tool, but almost impossible for an auto-regressive LLM even with retrieval augmentation as the input sequences are too specific to be indexed by a search engine.

3.4.5 Other implementations

Replacing Codex with gpt-3.5-turbo-16k (GeneGPT-turbo) achieves comparable overall performance (0.78 versus 0.83), showing that the turbo model can also use NCBI Web APIs with proper demonstrations. However, the performance of LangChain ReAct implementation (GeneGPT-lang) is lower, mainly because of E1 errors (described in §4.3) on Gene name conversion. Nonetheless, GeneGPT-lang still outperforms New Bing by 22% (0.54 versus 0.44).

Although evaluated under a more strict setting (§3.3), GeneGPT(-slim) achieves a macro-average performance of 0.83 which is much higher than other compared LLMs including New Bing (0.44). Overall, GeneGPT achieves new SOTA performance on all 2 one-shot tasks and 6 out of 7 zero-shot tasks and is outperformed by New Bing only on the gene disease association task.

4 Discussions

We have shown that GeneGPT largely surpasses various LLMs on GeneTuring. In this section, we further characterize GeneGPT by studying four research questions (RQ):

RQ1: What is the importance of each prompt component in GeneGPT?
RQ2: Can GeneGPT answer multi-hop questions by chain-of-thought API calls?
RQ3: What types of errors does GeneGPT make on each studied task?
RQ4: What is the scope of the questions that GeneGPT can answer?

4.1 RQ1: component importance

We conduct ablation and probing experiments to study the importance of individual prompt components, including two documentations (Dc.1, Dc.2) and four demonstrations (Dm.1–4) described in §2.2.

For ablation tests, we remove each component from GeneGPT-full and then evaluate the prompt. The results are shown in Fig. 3 (left). Notably, the performance on the DNA to genome and species alignment tasks is only significantly decreased without the BLAST demonstration (Dm.4), but not affected by the ablation of the BLAST documentation (Dc.2). While the ablations of other components decrease the performance, most only affect one relevant task (e.g. Dm.1 and gene name conversion), which indicates a high level of redundancy of the prompt components.

For the probing experiments, we evaluate GeneGPT with only one prompt component to study the individual capability. The results are shown in Fig. 3 (right). Overall, GeneGPT with only one documentation (Dc.1 or Dc.2) fails on all tasks. Surprisingly, with only one demonstration of the gene alias task (Dm.1) in the prompt, GeneGPT is able to perform comparably to GeneGPT-full on all tasks except the alignment ones. On the other hand, GeneGPT with only the BLAST demonstration (Dm.4) performs well on the two alignment tasks, which is somehow expected. These results suggest that GeneGPT with only two demonstrations (Dm.1 and Dm.4) in the prompt can

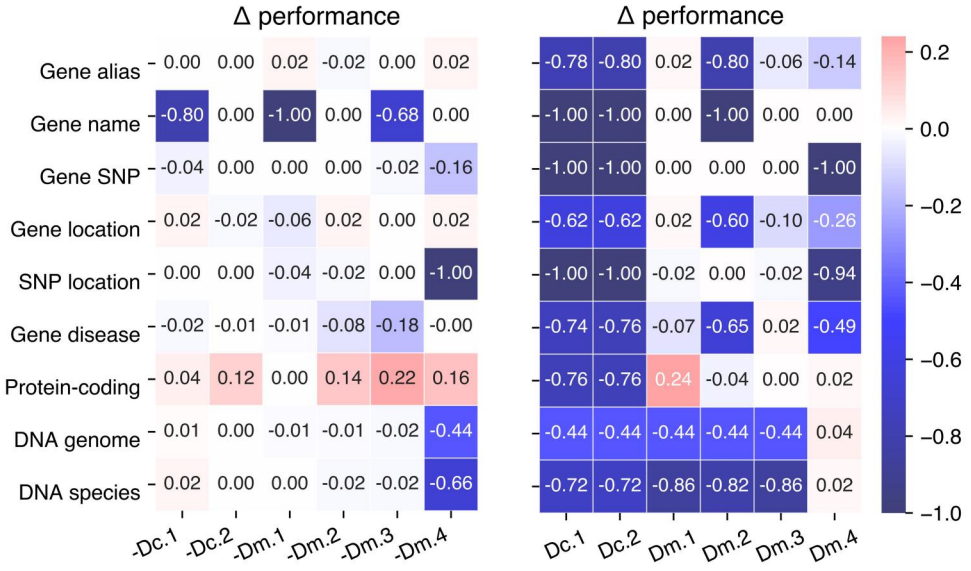

Figure 3. Performance changes of the ablation (left) and probing (right) experiments as compared to GeneGPT-full

generalize to all tasks in the GeneTuring benchmark. Table 2 show that with only two demonstrations (GeneGPT-slim), it outperforms the GeneGPT-full and achieves SOTA results on GeneTuring.

4.2 RQ2: multi-hop QA on GeneHop

Questions in the GeneTuring benchmark are single-hop and just require one step of reasoning, e.g. "Which gene is SNP rs983419152 associated with?". However, many real-world biomedical questions are multi-hop that need more steps to answer (Jin *et al.* 2022). For example, to answer "What is the function of the gene associated with SNP rs983419152?," the model should first get the associated gene name and then find its functions.

To test GeneGPT's capability of answering multi-hop questions, we present GeneHop, a novel dataset that contains three new multi-hop QA tasks based on the GeneTuring benchmark: (i) SNP gene function, which asks for the function of the gene associated with a given SNP. (ii) Disease gene location, where the task is to list the chromosome locations of the genes associated with a given disease. (iii) Sequence gene alias, which asks for the aliases of the gene that contains a specific DNA sequence. Each task in GeneHop contains 50 questions, and the collection pipeline is detailed in Supplementary Appendix C. For all tasks, we append the chain-of-thought instruction "Let's decompose the question to sub-questions and solve them step by step." after the test question (Wei *et al.* 2022b).

Figure 4 shows an example of GeneGPT to answer Task (i). Here, GeneGPT successfully decomposes the multi-hop question into two sub-questions, and the sub-question 2 is based on the answer of the sub-question 1. Interestingly, GeneGPT uses a shortcut to answer sub-question 2: instead of first calling `esearch` and then calling `esummary`, GeneGPT finds the gene id in the API call results of sub-question 1 and directly calls `esummary`. This capability is not shown in the prompt but elicited by chain-of-thought API calls.

Figure 5 shows another example of GeneGPT answering Task (ii), where GeneGPT successfully decomposes the multi-hop question and correctly calls the required APIs. Notably,

GeneHop question (SNP gene function):
What is the function of the gene associated with SNP rs1241371358? Let's decompose the question to sub-questions and solve them step by step.
GeneGPT output:
Sub-question 1: Which gene is SNP rs1241371358 associated with?
[https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi?db=snp&retmax=10&retmode=json&id=1241371358]
->[API call results]
Answer: LRRC23
Sub-question 2: What is the function of LRRC23?
[https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi?db=gene&retmax=10&retmode=json&id=10233]
->[API call results]
Answer: Predicted to be active in cytosol. (correct)

Figure 4. GeneGPT uses chain-of-thought API calls to answer a multi-hop question in GeneHop using a shortcut

the answering chain involves three sub-questions and four API calls, which are longer than all in-context demonstrations (one single-hop question and two API calls at most). This ability to generalize to longer chains of thought is an important aspect of GeneGPT's flexibility and usefulness for real-world applications.

We manually evaluate the results predicted by GeneGPT and compare it to the new Bing, which is the only baseline LLM that performs well on the single-hop GeneTuring benchmark due to its retrieval augmentation feature. The evaluation criteria are described in Supplementary Appendix D. As shown in Table 3, while the new Bing outperforms GeneGPT on the disease gene location task, it is mostly using webpages that contain both the disease and location information without multi-hop reasoning. The new Bing fails to perform the other two tasks since the input information (SNP or sequence) is not indexed by Bing and can only be found in specialized databases. GeneGPT, on the other hand, performs moderately well on all three tasks, and achieves a much higher average score (0.50 versus 0.24).

4.3 RQ3: error analysis

For error analysis, we manually study all mistakes made by GeneGPT and classify them into five types. Table 4 shows the count of each error type on the evaluate tasks: **E1:** using the wrong API or not using APIs, e.g. using the `gene` instead of the `omim` database for diseases; **E2:** using the right API but wrong arguments, e.g. passing terms to `id`; **E3:** not extracting the answer in the API result, most commonly seen in gene function extraction; **E4:** right API call but results do not contain the answer, where the question is not answerable with NCBI databases; and **O** includes other unclassified errors. Specific error examples are shown in Supplementary Appendix E.

Our results suggest that different tasks have specific and enriched error types: simple tasks (alias and location) fail mostly because of **E4**; **E1** only happens in disease-related tasks; alignment tasks face more issues with BLAST interfaces and reference genomes (**O**); multi-hop tasks in GeneHop tend to have **E2** and **E3** in the reasoning chains.

4.4 RQ4: Scope of usage

GeneGPT augments LLMs with NCBI Web APIs, so it is suitable for answering questions that require the use of biomedical database utilities, such as looking up the properties of a specific gene or SNP benchmarked by the GeneTuring dataset. GeneGPT can also answer complex questions requiring the sequential use of such functionalities, and this capability has been demonstrated by its performance on the GeneHop dataset. Formally, let $\mathcal{Q}$ be the set of single-hop questions that can be directly answered by an NCBI Web API endpoint. If a question X can be decomposed into a list of sub-questions $X \leftarrow [q_1, q_2, \ldots, q_i]$ where all $q_i \in \mathcal{Q}$, then the question can be theoretically answered by GeneGPT given the correct question decomposition. While GeneGPT has shown good performance on nine singe-hop tasks on GeneTuring and three multi-hop tasks on GeneHop, it is imperative to develop more diverse benchmark datasets and evaluate GeneGPT on them in the future. It should also be noted that GeneGPT is a model-agnostic augmentation method that can be applied to any backbone LLM without task-specific training. We believe not requiring any training is also an advantage of our augmentation strategy as it improves the

GeneHop question (Disease gene location):
List chromosome locations of the genes related to Cleft palate with ankyloglossia. Let's decompose the question to sub-questions and solve them step by step.
GeneGPT output:
Sub-question 1: What is the OMIM id of Cleft palate with ankyloglossia?
[https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=omim&retmax=20&retmode=json&sort=relevance&term=Cleft+palate+with+ankyloglossia]->[API call results]
Answer: 303400
Sub-question 2: What are genes related to Cleft palate with ankyloglossia?
[https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi?db=omim&retmax=20&retmode=json&id=303400]
->[API call results]
Answer: CPX
Sub-question 3: What is the chromosome location of CPX?
[https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=gene&retmax=20&retmode=json&sort=relevance&term=CPX]->[API call results]
[https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi?db=gene&retmax=20&retmode=json&id=64877,1377,7758276,100818277,100166185,10948718,10849362,9700326,3180310,3180309,1192676,1371,50945,10682,12892,1037,4336951,100216262,380161,108700304]
->[API call results]
Answer: Xq21.1 (correct)

Figure 5. GeneGPT uses long chain-of-thought of four API calls to answer a multi-hop question in GeneHop

Table 3. Performance of multi-hop QA on GeneHop.

GeneHop task	New bing	GeneGPT
SNP gene function	0.00	**0.55**
Disease gene location	**0.71**	0.67
Sequence gene alias	0.00	**0.28**
Average	0.24	**0.50**

We only compare GeneGPT with New Bing here since other LLMs cannot even answer single-hop questions well.
Bolded values denote the highest performance for the task.

backbone LLM's performance on various tasks without losing its generalizability.

5 Related work

5.1 Large language models

Recent studies have shown that scaling pre-trained LMs leads to performance improvement and potentially emergent abilities on various NLP tasks (Brown *et al.* 2020, Kaplan *et al.* 2020, Wei *et al.* 2022a, Chowdhery *et al.* 2022, OpenAI 2023). However, such auto-regressive LLMs are still susceptible to hallucinations and generate erroneous content (Ji *et al.* 2023). Augmenting LLMs with external tools is a possible solution to this issue (Mialon *et al.* 2023), which is the focus of this study.

Table 4. Counts of GeneGPT errors on different tasks.

GeneTuring Task	E1	E2	E3	E4	O
Gene alias	0	0	2	**6**	0
Gene location	0	0	0	**17**	0
SNP location	0	**1**	0	0	0
Gene disease association	**15**	0	0	3	2
DNA to human genome	0	0	7	0	**42**
DNA to multiple species	0	0	1	1	**4**
GeneHop Task	**E1**	**E2**	**E3**	**E4**	**O**
SNP gene function	0	0	**29**	0	0
Disease gene location	4	**7**	1	5	1
Sequence gene alias	0	**30**	8	0	0

E1: wrong API; E2: wrong arguments; E3: wrong comprehension; E4: unanswerable with API; O: others.
Bolded values denote the highest performance for the task.

5.2 Tool augmentation

Potential tools include: (i) search engines (Guu *et al.* 2020, Lewis *et al.* 2020, Borgeaud *et al.* 2022), also known as retrieval augmentation, exemplified by New Bing which is the previous SOTA on GeneTuring; (ii) program APIs by in-context learning (Gao *et al.* 2022, Schick *et al.* 2023) or fine-tuning (Parisi *et al.* 2022, Schick *et al.* 2023). In this work, we present the first study on the in-context learning capabilities of documentations and demonstrations of NCBI Web APIs.

5.3 Biomedical question answering

It is an essential step in clinical decision support (Ely *et al.* 2005) and biomedical knowledge acquisition (Jin *et al.* 2022). LLMs have been successfully applied to various biomedical QA tasks that are *knowledge*- or *reasoning*-intensive (Singhal *et al.* 2022, Liévin *et al.* 2022, Nori *et al.* 2023). However, auto-regressive LLMs fail to perform *data*-intensive tasks which require the model to precisely store and recite database entries, such as the GeneTuring benchmark (Hou and Ji 2023). Retrieval augmentation also falls short since specialized databases are usually not indexed by commercial search engines. GeneGPT solves this task with augmentations from domain-specific tools such as database utilities.

6 Conclusion

We present GeneGPT, a novel method that teaches LLMs to use NCBI Web APIs. It achieves SOTA performance on eight GeneTuring tasks and can perform chain-of-thought API calls. Our results indicate that database utility tools might be superior to relevant web pages for augmenting LLMs to faithfully serve various biomedical information needs.

Supplementary data

Supplementary data are available at *Bioinformatics* online.

Conflict of interest

None declared.

Funding

This research was supported by the NIH Intramural research, National Library of Medicine.

Data availability

The code and data used in this study is available at https://github.com/ncbi/GeneGPT.

References

Altschul SF, Gish W, Miller W *et al.* Basic local alignment search tool. *J Mol Biol* 1990;**215**:403–10.

Boratyn GM, Camacho C, Cooper PS *et al.* Blast: a more efficient report with usability improvements. *Nucleic Acids Res* 2013; **41**:W29–W33.

Borgeaud S, Mensch A, Hoffmann J *et al.* Improving language models by retrieving from trillions of tokens. In: *International conference on machine learning, Baltimore, Maryland, USA*, p. 2206–40. PMLR, 2022.

Brown T, Mann B, Ryder N *et al.* Language models are few-shot learners. *Advances in Neural Information Processing Systems* 2020; **33**:1877–901.

Chen M, Tworek J, Jun H *et al.* Evaluating large language models trained on code. arXiv, arXiv:2107.03374, 2021, preprint: not peer reviewed.

Chowdhery A, Narang S, Devlin J *et al.* Palm: scaling language modeling with pathways. arXiv, arXiv:2204.02311, 2022, preprint: not peer reviewed.

Ely JW, Osheroff JA, Chambliss ML *et al.* Answering physicians' clinical questions: obstacles and potential solutions. *J Am Med Inform Assoc* 2005;**12**:217–24.

Gao L, Madaan A, Zhou S *et al.* Pal: program-aided language models. arXiv, arXiv:2211.10435, 2022, preprint: not peer reviewed.

Guu K, Lee K, Tung Z *et al.* Retrieval augmented language model pre-training. In: *International conference on machine learning*, p. 3929–3938. PMLR, 2020.

Hou W, Ji Z. Geneturing tests gpt models in genomics. bioRxiv 2023: 2023–03. pages

Ji Z, Lee N, Frieske R *et al.* Survey of hallucination in natural language generation. *ACM Comput Surv* 2023;**55**:1–38.

Jin Q, Leaman R, Lu Z. Retrieve, summarize, and verify: how will chatgpt impact information seeking from the medical literature? *J Am Soc Nephrol* 2023a;**34**:1302–4.

Jin Q, Wang Z, Floudas CS *et al.* Matching patients to clinical trials with large language models. arXiv, arXiv:2307.15051, 2023b, preprint: not peer reviewed.

Jin Q, Yuan Z, Xiong G *et al.* Biomedical question answering: a survey of approaches and challenges. *ACM Comput Surv* 2022;**55**:1–36.

Kaplan J, McCandlish S, Henighan T *et al.* Scaling laws for neural language models. arXiv, arXiv:2001.08361, 2020, preprint: not peer reviewed.

Lewis P, Perez E, Piktus A *et al.* Retrieval-augmented generation for knowledge-intensive nlp tasks. *Adv Neural Inform Process Syst* 2020;**33**:9459–74.

Liévin V, Hother CE, Winther O. Can large language models reason about medical questions? arXiv, arXiv:2207.08143, 2022, preprint: not peer reviewed.

Luo R, Sun L, Xia Y *et al.* Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Brief Bioinform* 2022; **23**. https://doi.org/10.1093/bib/bbac409.

Mialon G, Dessì R, Lomeli M *et al.* Augmented language models: a survey. arXiv, arXiv:2302.07842, 2023, preprint: not peer reviewed.

Nori H, King N, McKinney SM *et al.* Capabilities of gpt-4 on medical challenge problems. arXiv, arXiv:2303.13375, 2023, preprint: not peer reviewed.

OpenAI. GPT-4 technical report. *CoRR* abs/2303.08774, 2023.

Parisi A, Zhao Y, Fiedel N. Talm: tool augmented language models. arXiv, arXiv:2205.12255, 2022, preprint: not peer reviewed.

Qin Y, Hu S, Lin Y *et al.* Tool learning with foundation models. arXiv, arXiv:2304.08354, 2023, preprint: not peer reviewed. http://arxiv.org/pdf/2304.08354.pdf.

Radford A, Narasimhan K, Salimans T *et al.* Improving language understanding by generative pre-training, OpenAI Blog, 2018.

Radford A, Wu J, Child R *et al.* Language models are unsupervised multitask learners. *OpenAI Blog* 2019;**1**:9.

Sayers EW, Agarwala R, Bolton EE *et al.* Database resources of the national center for biotechnology information. *Nucleic Acids Res* 2019;**47**:D23–D28.

Schick T, Dwivedi-Yu J, Dessì R *et al.* Toolformer: language models can teach themselves to use tools. arXiv, arXiv:2302.04761, 2023, preprint: not peer reviewed.

Schuler G, Epstein J, Ohkawa H *et al.* Entrez: molecular biology database and retrieval system. *Methods Enzymol* 1996;**266**:141–62.

Singhal K, Azizi S, Tu T *et al.* Large language models encode clinical knowledge. arXiv, arXiv:2212.13138, 2022, preprint: not peer reviewed.

Tian S, Jin Q, Yeganova L *et al.* Opportunities and challenges for chatgpt and large language models in biomedicine and health. *Brief Bioinform* 2024;**25**(1). https://doi.org/10.1093/bib/bbad493.

Wei J, Tay Y, Bommasani R *et al.* Emergent abilities of large language models. arXiv, arXiv:2206.07682, 2022a, preprint: not peer reviewed.

Wei J, Wang X, Schuurmans D *et al.* Chain of thought prompting elicits reasoning in large language models. arXiv, arXiv:2201.11903, 2022b, preprint: not peer reviewed.

Wong C, Zheng S, Gu Y *et al.* Scaling clinical trial matching using large language models: a case study in oncology. arXiv, arXiv:2308.02180, 2023, preprint: not peer reviewed.

Yao S, Zhao J, Yu D *et al.* React: synergizing reasoning and acting in language models. arXiv, arXiv:2210.03629, 2022, preprint: not peer reviewed.

Yuan J, Tang R, Jiang X *et al.* Llm for patient-trial matching: privacy-aware data augmentation towards better performance and generalizability. arXiv, arXiv:2303.16756, 2023, preprint: not peer reviewed.