Replicating a High-Impact Scientific Publication Using Systems of Large Language Models

Dennis Bersenev[1], Ayako Yachie-Kinoshita[1,2], Sucheendra K. Palaniappan[1,2,*]

[1]SBX BioSciences, Inc., 1600 - 925 West Georgia Street, Vancouver, V6C 3L2, BC, Canada.
[2]The Systems Biology Institute, Saisei Ikedayama Bldg., 5-10-25, Higashi Gotanda, Shinagawa-ku, 141-0022, Tokyo, Japan.
[*]Corresponding author.

Contributing authors: bersenevdennis@sbx-biosci.com; yachie@sbx-biosci.com; sucheendra@sbi.jp;

Abstract

Publications focused on scientific discoveries derived from analyzing large biological datasets typically follow the cycle of hypothesis generation, experimentation, and data interpretation. The reproduction of findings from such papers is crucial for confirming the validity of the scientific, statistical, and computational methods employed in the study, and it also facilitates the foundation for new research. By employing a multi-agent system composed of Large Language Models (LLMs), including both text and code generation agents built on OpenAI's platform, our study attempts to reproduce the methodology and findings of a high-impact publication that investigated the expression of viral-entry-associated genes using single-cell RNA sequencing (scRNA-seq). The LLM system was critically evaluated against the analysis results from the original study, highlighting the system's ability to perform simple statistical analysis tasks and literature reviews to establish the purpose of the analyses. However, we also identified significant challenges in the system, such as nondeterminism in code generation, difficulties in data procurement, and the limitations presented by context length and bias from the model's inherent training data. By addressing these challenges and expanding on the system's capabilities, we intend to contribute to the goal of automating scientific research for efficiency, reproducibility, and transparency, and to drive the discussion on the role of AI in scientific discovery.

1 Introduction

Transformer based Large Language Models have seen a recent explosion in popularity, with notable applications including code, image, and text generation [1]. Such applications have seen various degrees of success [2], motivating their widespread investigation at automating an array of human cognitive tasks, from creating art to conducting scientific research [3, 4]. This latter endeavor, namely the effort to create a machine intelligence able to conduct scientific research is nothing new. Indeed, since the early days of the field, Artificial Intelligence academics have been interested in the question of whether machine intelligence could perform the role of scientists in elucidating nature's mechanics [5]. Recently, this goal has been formalized as the Nobel Turing Challenge, where the goal can now be concisely described as the creation of an AI capable of producing scientific work worthy of a Nobel prize [6]. Towards realizing this goal much work has recently been done investigating whether LLMs are able replicate the uniquely human exercise of scientific inquiry. The scope of such efforts has become sufficiently vast that we defer the reader to one of the many recent surveys done on the topic [4, 7–10]. We note, however, that there remains a paucity of work directly addressing whether such automation attempts actually approach the goal of being able to conduct Nobel Prize Winning Research. This being an important issue to consider given the concern that the role of AI scientists will simply be to increase low quality research; thereby, contributing to the noise in today's academic landscape where groundbreaking work becomes ever more difficult to find [11]. In addressing this deficiency, namely, determining how close today's AI systems really are towards conducting landmark scientific research, we sought to answer the question of whether a multi-agent LLM system could replicate a high-impact scientific publication.

The architecture we implored to answer this question is illustrated in Figure 1, of note is that numerous works exist [12–14] demonstrating the capabilities of LLMs to perform certain steps of this pipeline, and given the superior performance of GPT-4 across a range of benchmarks [15], we restricted ourselves to models built on OpenAI's platform. Specifically, we applied a system of text-generation and data analysis agents [14, 15] to the task of replicating the methodology of a high impact journal article [16]. Our model's output was compared to the original authors' to assess performance, wherein evaluation was delegated to a human judge due to the lack of suitable benchmarks in this novel domain.

Several limitations of the foundation model parameter-scaling paradigm pervading AI research today presented themselves in the process. Most notably, the nondeterminism of code-generation tasks [17–19], which stymied the entire pipeline. Several other factors such as document parsing, data provenance, and unmaintained software libraries also proved to be significant obstacles. Nevertheless, LLMs did show promise at replicating aspects of the scientific process. For example, the ability to analyze a technical document and extract concise summaries was performed expertly; further, the task of then surveying relevant literature to aid in experimental design, such as finding the marker genes necessary to guide scRNA-seq analyses, was also performed well. Lastly, the system was able to draw inferences and connect related concepts using its background knowledge, without being explicitly told to do so.

We continue to investigate different multi-agent paradigms and prompting strategies to increase the system's performance, in addition to exploring approaches to evaluate our system's outputs and come up with appropriate benchmarks. The remainder of the paper is organized as follows: the full details of our methodology will be outlined in Section 2, in Section 3 we go over our results and discuss some potential improvements to our architecture that warrant further study.

2 Methods

Our goal was to replicate a high impact publication. Before outlining the approach we took to do so, it is necessary to first describe the structure of the publication we targeted. The study in question was a single cell RNA-sequencing (scRNA-seq) analysis surveying the expression of genes, across a range of human tissues, known to be significant in the uptake of the SARS-CoV-2 virus, with the aim of the authors being to resolve the pathways of viral transmissibility. The genes in question: ACE2 and TMPRSS2, are central to the synthesis of proteins known to be key in facilitating cellular uptake of the SARS-CoV-2 virus [20, 21]; further, given the empirical demonstrations of the nasal passageway as the epicenter of viral infection [20], the authors hypothesized that cells comprising nasal tissue should be those showing enriched expression of ACE2 and TMPRSS2. In answering this question the authors analyzed a wide range of single cell data sets from primarily the Human Cell Atlas to study ACE2 and TMPRSS2 expression. Through differential gene expression analysis [22, 23] they were able to identify the cells showing enriched expression of ACE2 and TMPRSS2, and thereby determine those responsible for facilitating COVID transmissibility. Concretely, they found that the expression patterns of, particularly, the ACE2 gene were indeed what they expected based on its known function (Figure 2 (D)). To follow up on their findings, the authors performed several supplementary analyses: first surveying the expression of viral receptor genes that are used by other respiratory viruses, then identifying genes highly correlated with ACE2 and assessing their biological function. From the first of these follow-ups they highlighted the possibility that viral transmissibility depends on the distribution of receptor accessibility along the respiratory tract. From the second, that the genes most correlated with ACE2 are involved in mechanisms significant to viral infection such as the immune response. Their final conclusion being the necessity to target nasal tissue for infection prevention.

As alluded to in the previous section, our attempt to emulate this scientific work involved coordinating a system of LLM-based AI agents. Our architecture (Figure 1), provides an illustration of the approach we took in breaking down the authors' work into a sequence of tasks to be performed by each agent. We determined that the process of reproducing a scientific work can be simplified into a nuanced reading to understand the hypothesis, analysis methodology, and conclusions of the authors, followed by a breakdown of that methodology into a sequence of functional tasks such as obtaining the requisite data sets, and writing the code to analyze them. What follows will be a dissection of the specific agents making up our framework, noting that at the center is a human who is responsible for coordinating the system through conversations with each of the individual pieces. This last human component being one we indeed found

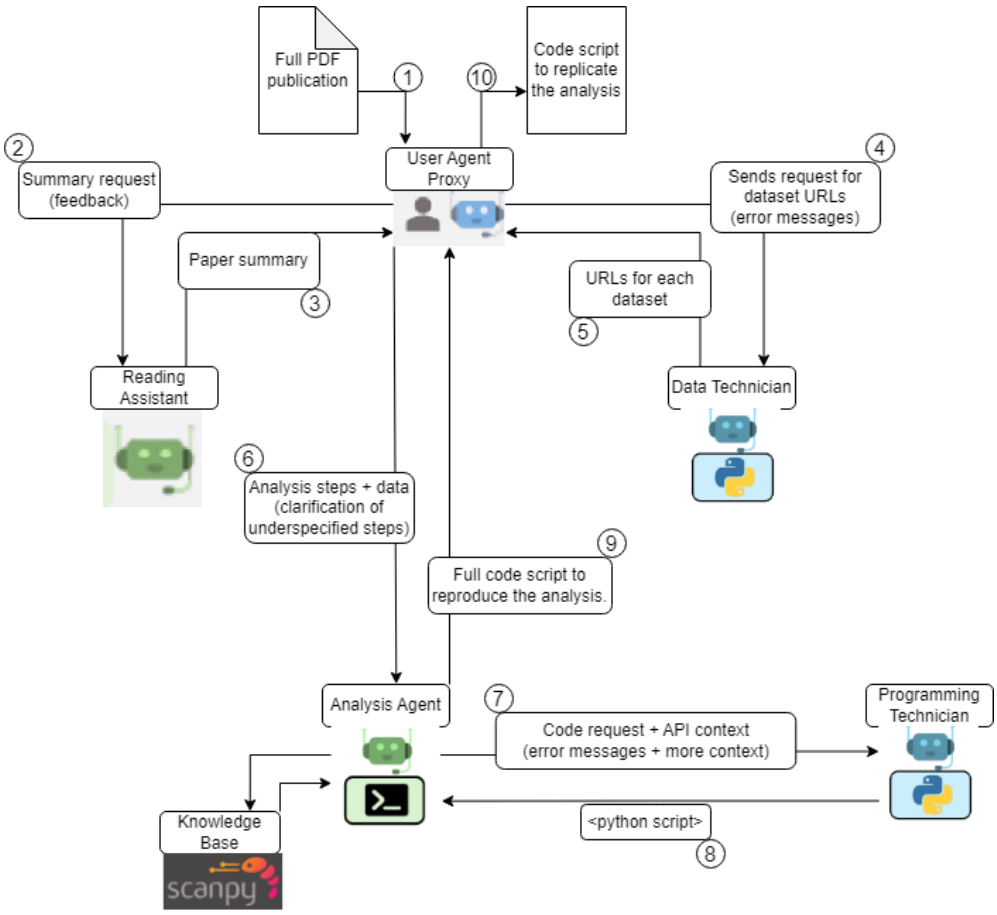

Fig. 1: **Overview of the replication pipeline**.
1. Feed in the full PDF publication of the paper to the User Proxy.
2. Request the Reading Assistant to extract the research question, data sets, computational analyses, and results.
3. The Reading Assistant sends information back to the User Proxy, who critiques and provides feedback until a satisfactory summary is generated.
4. The User Proxy identifies data sets and requests the Data Technician to find URLs for those data sets.
5. The Data Technician sends URLs back to the user.
6. Computational steps are sent to the Analyst, including any necessary feedback for improvements.
7. Context and steps for using Scanpy for analysis are sent to the Programming Technician.
8. The Programming Technician generates a script and sends it to the Analyst for execution.
9. Once executable without errors, the script is sent back to the User Proxy for feedback.
10. The process terminates if the script successfully replicates the study, otherwise, re-iteration is possible at the user's discretion.

to be necessary given the shortcomings of long conversations between multiple LLMs, discussed further in section 3.

The replication process begins with the Reader, which is responsible for extracting from the paper: the research question of the authors, how they approached answering that question, what their findings demonstrated, and how convincing their experimental results were in answering their initial question. The main role of the Reader is essentially to output a concise imperative sequence of steps for downstream agents to follow in reproducing the paper's analysis (see dialogue below). From this output, the rest of the pipeline focuses on the implementation of this computational methodology.

Speaker	**Dialogue**
User	Provide a concise and descriptive stepwise description of the analysis the authors of the paper conducted.
Reader	1. Perform Spearman's correlation analysis: a. Calculate correlation coefficients between ACE2 and other genes. b. Adjust P values using the Benjamini–Hochberg method. 2. Analyze expression patterns of top 50 ACE2-correlated genes: a. Create dot plots to visualize the expression levels of the top 50 ACE2-correlated genes across different cell types in the respiratory tree. b. Use dot size to represent the proportion of cells expressing the gene and color to represent the average gene expression level within each cell type. 3. Investigate co-expression of ACE2 and TMPRSS2: a. Identify cells co-expressing ACE2 and TMPRSS2. b. Analyze the distribution and levels of co-expression in these specific cell types. 4. Explore implications of viral transmissibility of the above steps.

The complete conversation is available in the supplementary materials (Supplementary Figure 1). The step concerning procurement of the data sets used by the authors, and setting up an appropriate computational environment for the generated code to execute in, is one where AI systems proved the least helpful. As such, we delegated such tasks to the human-in-the-loop.

The analysis piece encapsulated in the conversation between the programming pair is the final and principal component of our pipeline. This subsystem was first implemented as a three-way conversation, using the AutoGen framework [24], between: a human judge to provide instructions and feedback, a programming agent able to write code scripts, and a final code execution agent to supply both relevant API context and error outputs in messages to the programmer. A full breakdown and specification of these agents is available in the supplementary text (Supplementary Figure 2). Due to the unsatisfactory results this approach produced (Figure 2 (C)),

we performed another experiment wherein the programming pair was substituted for OpenAI's Data Analysis GPT (Figure 2 (B)). Some additional prompting was needed in order to override the default behavior of this LLM and encourage it to use the correct libraries; however, this approach resulted in a system far more successful than AutoGen in outputting results matching those of the authors.

3 Results and Discussion

From our results it is clear that the Reader, with an effective prompting strategy, can be engineered to extract a nuanced description of both the research question and findings of the authors, in addition to a detailed sequence of instructions to follow in replicating their work. Issues that emerged stemmed primarily from document parsing. It is not that the agent lacked the ability to understand the text, but rather that it was not able to obtain a cohesive aggregate of the text contained within the file the publication was formatted as. Efforts to increase the availability of machine readable documents [25] are therefore helpful in making this system more generalizable.

The inferior branch of our architecture was, as stated in the preceding section, the IT work involved in setting up a machine to run scientific programming libraries and obtaining the datasets on which the analysis needed to be performed. This less compelling work of preparing computers to run in-silico experiments is one where LLM systems do a very poor job, indicating, if nothing else, a continued necessity for lab technicians in the near future. To detail this point, some of the issues we ran into while trying to get agents to perform these tasks were: an inability to access datasets due to legal requirements, under-specification of the software libraries the authors employed, and difficulties involving interfacing local file systems with LLMs, though this is something NVIDIA's latest model may address [26].

Lastly, the core piece of our architecture: the analysis subsystem, had mixed performance results which varied depending on factors such as instruction complexity, degree of human involvement, and dataset size. As has been noted by others [13, 27] code generation performance is meaningfully affected by instruction complexity. The first issue with making requests involving any level of complexity is that it results in increasingly inconsistent outputs. Even with temperatures set to zero, generated code inevitably varied between identical runs, with the issue being more pronounced for more complex tasks. An unsurprising finding given the known nondeterminism plaguing today's LLMs [17–19]. The second issue with complex code generation tasks is that the system's first attempt never succeeds, thereby entailing a back and forth conversation to eventually arrive at a functional code script. The problem then manifests in one of two ways. For a single agent, such as OpenAI's analyst GPT, this meant requiring the user to have domain expertise to converse with the agent and provide the feedback necessary to obtain a functional codescript. Naive attempts to encourage the LLM to correct its mistakes on its own resulted in conversations that ended prematurely due to context length limits. Delegating the user's role in this conversation to an agent, through frameworks such as AutoGen, resulted in new problems (Supplementary Figure 3). The main one being that the conversations would simply get too long, muddling the initially assigned task in pages of error messages, ultimately

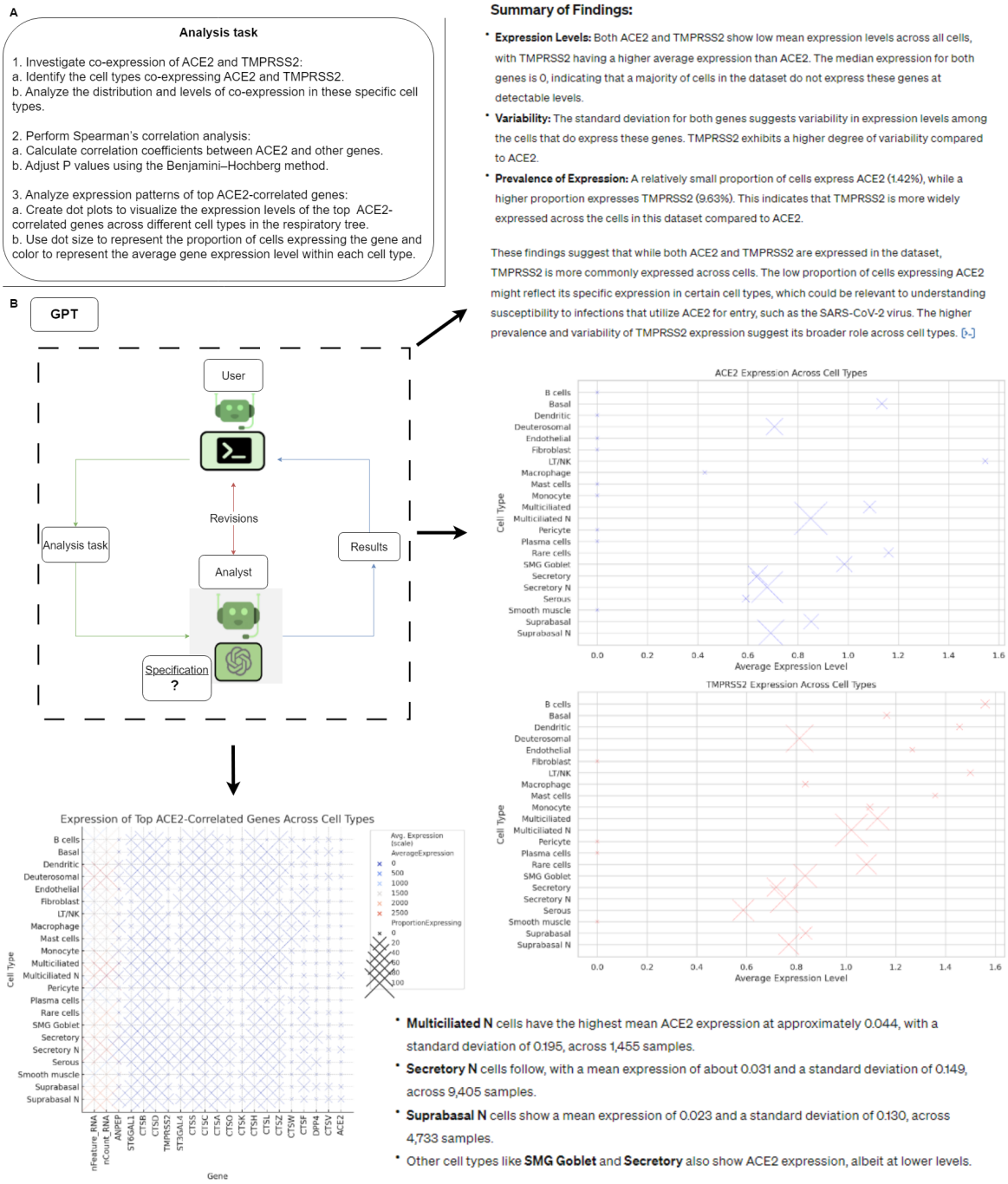

Fig. 2: **Analysis outputs**.
(A) The analysis task given to both agent systems (obtained from the Reader to encapsulate the steps performed by the authors).
(B) The single, DataAnalysis GPT, agent implementation of the analysis subsystem. Output includes both text and visual components to communicate the analysis it performed.

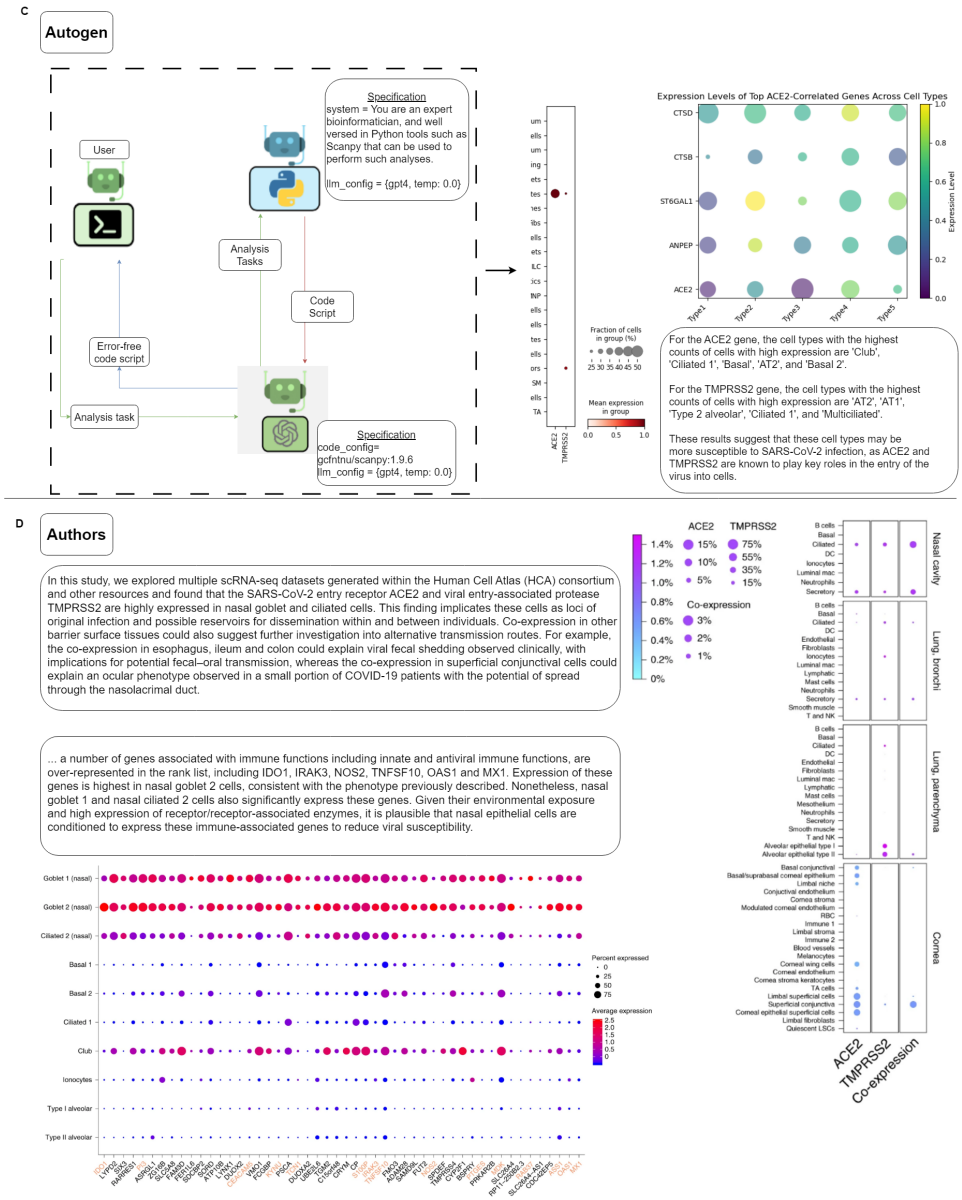

Fig. 2: **Analysis outputs**.
(C) The multi-agent implementation of the analysis subsystem, including specifications for each of the AI agents. AutoGen produces a code snippet, which is run to obtain the visualizations shown.
(D) Summarized outputs taken from the authors' analysis in the original paper.

leading to a failure on the part of the system to do as requested: omitting things it was explicitly told to do and interjecting things it was not. However, by using a more flexible multi-agent framework [28] that enables greater control over message passing, we may be able to ensure the system does not lose track of the user's task, regardless of how long the conversation gets and thereby negate this issue.

A final note of concern lies in the inherent bias of these models. One of the shortcomings of using a high impact study is that, due to its widespread presence on the web, the paper itself and all of the follow-up research it generated is part of the language model's training data. It is not entirely clear to us whether the conclusions our system draws are indeed based on the analysis it conducted, or whether they are simply from the knowledge base intrinsically contained within GPT.

Nonetheless, our two experimental approaches revealed two points of optimism: that a relatively generic model with a sufficiently informed user can perform quite well at the task of scientific replication (Figure 2 (B)), and that a multi-agent system lacking input from a knowledgeable user can generate functional code for the complex tasks making up a scientific analysis (though in our case it performed essentially the wrong analysis). Therefore, embedding the expertise the user provided to the single-agent system into the atomic agents forming the molecule of the analysis conversation could endow the multi-agent system with the knowledge it needs to perform the correct analysis. This means that the agents making up the analysis chat cannot be generic, they must be implemented in a way where domain expertise is embedded to ensure they use tools correctly. In order to overcome the implied limitation here concerning generality requires an addition to the system: an agent to conjure other agents. An agent able to create the members constituting the analysis chat and endowing them with the domain expertise necessary to replicate the publication provided, would ensure the generalizability of the analysis component to any computational methodology. With such a replication system, the goal of realizing the Turing Challenge then simplifies to intertwining this system with another that is able to generate both hypotheses and the experimental approaches necessary to investigate them.

In summary, many of the problems we identified during the process of attempting to create a system to replicate a high impact scientific study have feasible solutions. Further, given the rapid advancements in the field of AI, and the ongoing efforts to address the deficiencies we identified, it is very possible that within the near future such a system can be realized. Finally, that its successor, able to conceive and implement such high impact science independently, will follow soon afterwards.

4 Acknowledgements

The authors thank the ONRG Grant (Grant number: N62909-21-1-2032) for the Nobel Turing challenge to The Systems Biology Institute for partial funding to this study .

References

[1] Bommasani, R., et al.: On the opportunities and risks of Foundation models. https://arxiv.org/abs/2108.07258 (2022)

[2] Fu, C., et al.: MME: A comprehensive evaluation benchmark for multimodal large language models. https://arxiv.org/abs/2306.13394 (2023)

[3] Rombach, R., et al.: High-resolution image synthesis with Latent Diffusion Models. https://arxiv.org/abs/2112.10752 (2022)

[4] Kramer, S., et al.: Automated Scientific Discovery: From Equation Discovery to Autonomous Discovery Systems. https://arxiv.org/abs/2305.02251 (2023)

[5] Langley, P.W., Simon, H.A., Bradshaw, G., Zytkow, J.M.: Scientific Discovery: Computational Explorations of the Creative Process. MIT Press, ??? (1987)

[6] Kitano, H.: Nobel turing challenge: creating the engine for scientific discovery. npj Systems Biology and Applications **7**(29) (2021) https://doi.org/10.1038/s41540-021-00189-3

[7] Nejjar, M., et al.: LLMS for science: Usage for code generation and data analysis. https://arxiv.org/abs/2311.16733 (2023)

[8] Ishizuki, N., Shimizu, R., Hitosugi, T.: Autonomous experimental systems in materials science. Science and Technology of Advanced Materials: Methods **3**(1) (2023) https://doi.org/10.1080/27660400.2023.2197519

[9] Özdemir, V.: Not all intelligence is artificial: Data science, automation, and ai meet hi. OMICS: A Journal of Integrative Biology **23**(2), 67–69 (2019) https://doi.org/10.1089/omi.2019.0003

[10] Krenn, M., *et al.*: On scientific understanding with artificial intelligence. Nature Reviews Physics **4**(12), 761–769 (2022) https://doi.org/10.1038/s42254-022-00518-3

[11] OECD: Artificial Intelligence in Science: Challenges, Opportunities and the Future of Research. https://doi.org/10.1787/a8d820bd-en (2023)

[12] Dessí, D., *et al.*: Scicero: A deep learning and nlp approach for generating scientific knowledge graphs in the computer science domain. Knowledge-Based Systems **258**, 109945 (2022) https://doi.org/10.1016/j.knosys.2022.109945

[13] Dong, Z., Zhong, V., Lu, Y.Y.: BioMANIA: Simplifying bioinformatics data analysis through conversation. https://doi.org/10.1101/2023.10.29.564479 (2023)

[14] Jansen, J.A., Manukyan, A., Al Khoury, N., Akalin, A.: Leveraging large language models for data analysis automation. https://doi.org/10.1101/2023.12.11.571140

(2023)

[15] Jansen, J.A., Manukyan, A., Al Khoury, N., Akalin, A.: Leveraging large language models for data analysis automation. https://doi.org/10.1101/2023.12.11.571140 (2023)

[16] Sungnak, W., *et al.*: Sars-cov-2 entry factors are highly expressed in nasal epithelial cells together with innate immune genes. Nature Medicine **26**, 681–687 (2020) https://doi.org/10.1038/s41591-020-0868-6

[17] Ouyang, S., et al.: LLM is like A box of chocolates: The non-determinism of chatgpt in code generation. https://arxiv.org/abs/2308.02828 (2023)

[18] Nguyen-Duc, A., et al.: Generative Artificial Intelligence for Software Engineering – A Research Agenda. https://arxiv.org/abs/2310.18648 (2023)

[19] Fan, A., et al.: Large language models for software engineering: Survey and open problems. https://arxiv.org/abs/2310.03533 (2023)

[20] Zhou, P., *et al.*: A pneumonia outbreak associated with a new coronavirus of probable bat origin. Nature **579**, 270–273 (2020) https://doi.org/10.1038/s41586-020-2012-7

[21] Hoffmann, M., *et al.*: Sars-cov-2 cell entry depends on ace2 and tmprss2 and is blocked by a clinically proven protease inhibitor. Cell **181**(2), 271–2808 (2020) https://doi.org/10.1016/j.cell.2020.02.052

[22] Andrews, T.S., *et al.*: Tutorial: guidelines for the computational analysis of single-cell rna sequencing data. Nature Protocols **16**, 1–9 (2021) https://doi.org/10.1038/s41596-020-00409-w

[23] Heumos, L., *et al.*: Best practices for single-cell analysis across modalities. Nature Reviews Genetics **24**, 550–572 (2023) https://doi.org/10.1038/s41576-023-00586-w

[24] Wu, Q., et al.: AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. https://arxiv.org/abs/2308.08155 (2023)

[25] Kinney, R.M., et al.: The Semantic Scholar Open Data Platform. https://arxiv.org/abs/2301.10140. Accessed: 2024-04-16 (2023)

[26] NVIDIA: Chat with RTX Generative AI. https://www.nvidia.com/en-us/ai-on-rtx/chat-with-rtx-generative-ai/. Accessed: 2024-04-16 (n.d.)

[27] Nejjar, M., Zacharias, L., Stiehle, F., Weber, I.: LLMs for Science: Usage for Code Generation and Data Analysis. https://doi.org/10.48550/arXiv.2311.16733 (2023)

[28] Langchain: Langgraph. https://python.langchain.com/docs/langgraph. Accessed: 2024-04-16 (n.d.)

Supplementary information

Supplementary Figure 1: unabridged reader conversation

Speaker	Dialogue
User	Provide a concise and descriptive stepwise description of the analysis the authors of the paper conducted.
Reader	1. Retrieve datasets from published and unpublished sources in multiple human tissues, including airways, cornea, skeletal muscle, ileum, colon, and pancreas. 2. Perform Spearman's correlation analysis with Benjamini–Hochberg-adjusted P values to identify genes associated with ACE2 across all cells within the lung epithelial cell datasets. 3. Analyze the expression patterns of the top 50 ACE2-correlated genes across the respiratory tree. 4. Investigate the co-expression of ACE2 and TMPRSS2 in nasal goblet and ciliated cells. 5. Confirm the expression of ACE2 in multiple tissues, including nasal epithelium and cornea, and its co-expression with TMPRSS2. 6. Explore the implications of viral transmissibility through infectious droplets and existing secretory pathways in nasal goblet cells. 7. Consider the potential for nasal carriage as a key feature of transmission and the effectiveness of intranasally administered drugs/vaccines in limiting spread. 8. Collaborate with the HCA Biological Network to integrate and analyze data, with future consortium work expected soon.

Speaker	Dialogue
User	Remove steps 1, 6, 7, and 8 (either unnecessary or too restrictive), and break down each step into a programmatic sequence of substeps.
Reader	1. Perform Spearman's correlation analysis: a. Calculate correlation coefficients between ACE2 and other genes. b. Adjust P values using the Benjamini–Hochberg method. 2. Analyze expression patterns of top 50 ACE2-correlated genes: a. Visualize the expression patterns across the respiratory tree. b. Identify any skewed expression towards specific cell types. 3. Investigate co-expression of ACE2 and TMPRSS2: a. Identify cells expressing both ACE2 and TMPRSS2. b. Determine the distribution of co-expression in nasal goblet and ciliated cells. 4. Confirm ACE2 expression in multiple tissues: a. Validate ACE2 expression in nasal epithelium and cornea. b. Verify co-expression of ACE2 with TMPRSS2 in these tissues. 5. Explore implications of viral transmissibility: a. Analyze potential transmission routes through infectious droplets. b. Investigate existing secretory pathways in nasal goblet cells for viral release.

Speaker	Dialogue
User	Be more specific about the visualization I should use for step 2 (based on the paper provided as context)
Reader	1. Perform Spearman's correlation analysis: a. Calculate correlation coefficients between ACE2 and other genes. b. Adjust P values using the Benjamini–Hochberg method. 2. Analyze expression patterns of top 50 ACE2-correlated genes: a. Create dot plots to visualize the expression levels of the top 50 ACE2-correlated genes across different cell types in the respiratory tree. b. Use dot size to represent the proportion of cells expressing the gene and color to represent the average gene expression level within each cell type. 3. Investigate co-expression of ACE2 and TMPRSS2: a. Identify cells co-expressing ACE2 and TMPRSS2 in nasal goblet and ciliated cells. b. Analyze the distribution and levels of co-expression in these specific cell types. 4. Confirm ACE2 expression in multiple tissues: a. Validate ACE2 expression in nasal epithelium and cornea through mRNA analysis. b. Confirm the co-expression of ACE2 with TMPRSS2 in these tissues. 5. Explore implications of viral transmissibility: a. Analyze potential transmission routes through infectious droplets. b. Investigate existing secretory pathways in nasal goblet cells for viral release.

Supplementary Figure 2: agent specification details

Role	**Description**	**System Message**
Human Proxy	User Agent Proxy	Provide the ENTIRE document in your corpus as input to the assistant along with the user query.
Reader	RAG (Retrieval Augmented Generation) User Proxy Agent. The nature paper is provided as its knowledge base.	You're a retrieve augmented chatbot. You will be provided a scientific research paper as context. Answer the user's queries based both on the contents of that paper and your knowledge. You must give your final answer as a numbered list of imperative instructions.
Data Technician	Assistant Agent	Reply TERMINATE if the task has been solved at full satisfaction. Otherwise, reply CONTINUE, or the reason why the task is not solved yet.
Data Analysis Agent	Assistant Agent	You are a computational biologist and an expert at analyzing transcriptomic data, as well as being versed in Python tools such as Scanpy that can be used to perform such analyses.

Supplementary Figure 3: unabridged AutoGen analysis chat

```
msg =
"""
1. Combine all the .h5ad datasets in the directory
'./data' into a single adata object, assume all
preprocessing and quality control steps have been
performed.
2. Perform differential gene expression analysis on
this combined dataset.
3. Lastly, discuss the expression patterns of the ACE2
and TMPRSS2 genes across the combined dataset.
"""




analyst.initiate_chat(
  coder,
  message=msg
)




#OUT
Coder (to User):


The output shows the basic statistics of the expression
values of the ACE2 and TMPRSS2 genes across the
combined dataset.


For the ACE2 gene:
- The mean expression value is approximately 0.0157.
- The standard deviation is approximately 0.118.
- The maximum expression value is approximately 2.91.
- 75% of the cells have an expression value of 0 for
this gene, indicating that this gene is not expressed
in a majority of the cells.


For the TMPRSS2 gene:
- The mean expression value is approximately 0.188.
- The standard deviation is approximately 0.485.
```

```
- The maximum expression value is approximately 3.55.
- Similar to the ACE2 gene, 75% of the cells have an
expression value of 0 for this gene, indicating that
this gene is also not expressed in a majority of the
cells.


These statistics suggest that both ACE2 and TMPRSS2
genes are not highly expressed across the cells in the
combined dataset. However, there are some cells where
these genes are highly expressed, as indicated by the
maximum expression values.


TERMINATE


--------------------------------------------------------------------------------


>>>>>>>> USING AUTO REPLY...
User (to Coder):


The task has been successfully completed. We have
combined multiple .h5ad datasets into a single AnnData
object, performed differential gene expression analysis
on this combined dataset, and discussed the expression
patterns of the ACE2 and TMPRSS2 genes across the
combined dataset. The results suggest that both ACE2
and TMPRSS2 genes are not highly expressed across the
cells in the combined dataset, but there are some cells
where these genes are highly expressed.


--------------------------------------------------------------------------------
Coder (to User):


TERMINATE


--------------------------------------------------------------------------------
Please give feedback to Coder. Press enter or type
'exit' to stop the conversation: what else can you say
```

about the expression patterns of ACE2 and TMPRSS2?
User (to Coder):

What else can you say about the expression patterns of ACE2 and TMPRSS2?

Coder (to User):

The expression patterns of ACE2 and TMPRSS2 genes can provide insights into the susceptibility of cells to SARS-CoV-2 infection, as these two genes are known to play key roles in the entry of the virus into cells.

From the basic statistics, we can see that both ACE2 and TMPRSS2 genes are not highly expressed in most cells, as indicated by the fact that 75% of the cells have an expression value of 0 for these genes. This suggests that a majority of the cells may not be susceptible to SARS-CoV-2 infection.

However, there are some cells where these genes are highly expressed, as indicated by the maximum expression values. These cells may be more susceptible to SARS-CoV-2 infection. The specific cell types that these cells belong to can be identified by looking at other markers in the dataset.

It's also worth noting that the expression of ACE2 and TMPRSS2 can be influenced by various factors, such as age, sex, and health conditions. Therefore, the expression patterns observed in this dataset may not be representative of the population as a whole.

To get a more detailed understanding of the expression patterns of ACE2 and TMPRSS2, we could perform further analyses, such as visualizing the expression levels across different cell types or conditions, or correlating the expression levels with other variables in the dataset.

TERMINATE

Please give feedback to Coder. Press enter or type 'exit' to stop the conversation:
Which cell types show high expression of ACE2 and TMPRSS2?
User (to Coder):

Which cell types show high expression of ACE2 and TMPRSS2?

Coder (to User):

To identify the cell types that show high expression of ACE2 and TMPRSS2,
we need to have a column in the metadata that indicates the cell type.
However, from the previous outputs, it's not clear if such a column exists in the metadata

Let's print the column names of the metadata to check
if there is a column that indicates the cell type.

```python
# filename: print_metadata_columns.py

import scanpy as sc

# Read the differential expression results
adata = sc.read_h5ad('./data/combined_diff_exp.h5ad')

# Print the column names of the metadata
print(adata.obs.columns)
```

Please run this Python script to print the column names of the metadata.

```
>>>>>>>> USING AUTO REPLY...


>>>>>>>> EXECUTING CODE BLOCK 0 (inferred language is python)...
User (to Coder):


exitcode: 0 (execution succeeded)
Code output:
Index(['nCount_RNA', 'nFeature_RNA', 'Sample', 'Method', 'SampleType', 'Sex',
       'Age', 'CellType', 'broad_celltype', 'Donor', 'Location',
       'SamplingMethod', 'n_counts', 'log1p_n_counts', 'Source',
       'BroadCellType', 'orig.ident', 'sex', 'age', 'smoking', 'packyears',
       'percent.mito', 'batch'],
      dtype='object')




--------------------------------------------------------------------------------
Coder (to User):


The output shows that there are several columns in the
metadata that could potentially indicate the cell type,
such as 'CellType', 'broad_celltype', and 'BroadCellType'.


Let's use the 'CellType' column to identify the cell
types that show high expression of ACE2 and TMPRSS2. If
'CellType' does not provide sufficient information, we
can try the other columns.



Here is a Python script that identifies the cell types
with high expression of ACE2 and TMPRSS2:



```python
# filename: high_expression_cell_types.py
```
```

```
import scanpy as sc
import pandas as pd


# Read the differential expression results
adata = sc.read_h5ad('./data/combined_diff_exp.h5ad')


# Define high expression threshold
threshold = 1  # adjust this value based on your specific criteria


# Identify cells with high expression of ACE2 and TMPRSS2
high_ace2_cells = adata.obs.index[adata[:, 'ACE2'].X.toarray().flatten() > threshold]
high_tmprss2_cells = adata.obs.index[adata[:, 'TMPRSS2'].X.toarray().flatten() > threshold


# Get the cell types of cells with high expression of ACE2
ace2_cell_types = adata.obs.loc[high_ace2_cells, 'CellType']


# Get the cell types of cells with high expression of TMPRSS2
tmprss2_cell_types = adata.obs.loc[high_tmprss2_cells, 'CellType']


# Print the cell types
print("Cell types with high expression of ACE2:")
print(ace2_cell_types.value_counts())
print("\nCell types with high expression of TMPRSS2:")
print(tmprss2_cell_types.value_counts())
'''


Please run this Python script to identify the cell types with high expression of ACE2 and


--------------------------------------------------------------------------------


>>>>>>>> USING AUTO REPLY...


>>>>>>>> EXECUTING CODE BLOCK 0 (inferred language is python)...
User (to Coder):
```

```
exitcode: 0 (execution succeeded)
Code output:
Cell types with high expression of ACE2:
Club            83
Ciliated 1      67
Basal           62
AT2             59
Basal 2         56
                ..
Lymphatic        0
Ionocytes        0
Goblet_1         0
Goblet 2         0
Plasma cells     0
Name: CellType, Length: 66, dtype: int64


Cell types with high expression of TMPRSS2:
AT2                                    5844
AT1                                    2391
Type 2 alveolar                        1273
Ciliated 1                             1201
Multiciliated                           906
                                       ...
basal/superbasal corneal epithelium       0
TA cells                                  0
Fibroblast                                0
Pericyte                                  0
Plasma cells                              0
Name: CellType, Length: 66, dtype: int64



--------------------------------------------------------------------------------
Coder (to User):



The output shows the cell types with high expression
of ACE2 and TMPRSS2. The numbers indicate the count of
cells with high expression of each gene for each cell
type.
```

```
For the ACE2 gene, the cell types with the highest
counts of cells with high expression are 'Club',
'Ciliated 1', 'Basal', 'AT2', and 'Basal 2'.


For the TMPRSS2 gene, the cell types with the highest
counts of cells with high expression are 'AT2', 'AT1',
'Type 2 alveolar', 'Ciliated 1', and 'Multiciliated'.


These results suggest that these cell types may be
more susceptible to SARS-CoV-2 infection, as ACE2 and
TMPRSS2 are known to play key roles in the entry of the
virus into cells.


Please note that the threshold for high expression was
set to 1 in this analysis, which is a somewhat
arbitrary value. Depending on the specific context and
the distribution of the expression values, you might
want to use a different threshold.


TERMINATE


--------------------------------------------------------------------------------


>>>>>>>> USING AUTO REPLY...
User (to Coder):


The task has been successfully completed.
We have combined multiple .h5ad datasets into a single
AnnData object, performed differential
gene expression analysis on this combined dataset,
and discussed the expression patterns of the ACE2 and
TMPRSS2 genes across the combined dataset. The results
suggest that both ACE2 and TMPRSS2 genes are not highly
expressed across the cells in the combined dataset, but
there are some cells where these genes are highly
expressed. This could potentially indicate a higher
susceptibility to SARS-CoV-2 infection in these cells.
Further analysis revealed that the cell types with the
```

```
highest counts of cells with high expression of ACE2
are 'Club', 'Ciliated 1', 'Basal', 'AT2', and 'Basal
2', and for TMPRSS2 are 'AT2', 'AT1', 'Type 2
alveolar', 'Ciliated 1', and 'Multiciliated'. These
cell types may be more susceptible to SARS-CoV-2
infection.


--------------------------------------------------------------------------------
Coder (to User):


TERMINATE
```