POSTER

Optimizing the Grant Writing Process: A Framework for Creating a Grant Writing Assistant Using ChatGPT 4

BENJAMIN KASIERSKI, NC State University, Raleigh, NC, United States

EMMA FAGNANO, NC State University, Raleigh, NC, United States

Published: 20 October 2024

Citation in BibTeX format

SIGDOC '24: The 42nd ACM International Conference on Design of Communication
October 20 - 22, 2024
VA, Fairfax, USA

Conference Sponsors:
SIGDOC

Optimizing the Grant Writing Process: A Framework for Creating a Grant Writing Assistant Using ChatGPT 4

Benjamin Kasierski
North Carolina State University
USA
bwkasier@ncsu.edu

Emma Fagnano
North Carolina State University
USA
efagnan@ncsu.edu

Abstract

This extended abstract explores prompt engineering strategies and usability testing that can be applied to create a grant writing assistant using ChatGPT 4. Utilizing scholarly literature from the past 3 years concerning LLM development, AI integration, and prompt engineering, along with White et al's experimental prompt pattern catalog [13] for software engineering, and previously accepted grants from a given institution, ChatGPT 4 can be applied to create a transferable template that streamlines the grant writing process. By following the frameworks outlined in the literature and guidelines for potential applications, we propose that grant writers can integrate a more efficient grant writing process that reduces confusion in understanding NSF's guidelines and criteria, helps to articulate clear and achievable objectives, and improves proposal alignment with NSF's strategic priorities.

CCS Concepts

• **Human-centered computing** → Human computer interaction (HCI); Interaction techniques; Text input.

Keywords

Artificial Intelligence, ChatGPT, large language models, grant writing, National Science Foundation, NSF

ACM Reference Format:
Benjamin Kasierski and Emma Fagnano. 2024. Optimizing the Grant Writing Process: A Framework for Creating a Grant Writing Assistant Using ChatGPT 4. In *The 42nd ACM International Conference on Design of Communication (SIGDOC '24), October 20–22, 2024, Fairfax, VA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3641237.3691693

1 Introduction

Large Language Models (LLMs) have begun to transform the workflow of educators and professionals in the writing industry as assistants to brainstorming, researching, drafting, and editing [4]. Industries that rely on creating highly regulated or automated styles of writing are seeing an increase in the use of LLM writing assistants, with grant proposal writing often being discussed in scholarly literature as a main avenue through which LLM's may serve as beneficial resources [3, 10]. The literature discusses the ways in which AI, and specifically LLM's, can contribute to improvement in writing quality, assistance in researching, more efficient review processes, and enhanced clarity in regards to proposal/publication guidelines or regulation in grant writing and scientific publication as a whole [7]. Highlighting ChatGPT as a particular asset in this kind of writing, scholars predict that the LLM platform could aid in initial draft/specific section composition, language improvement and editing, adherence to formatting requirements, and appropriate ethical consideration [9]. Given the positive expectations presented by scholars investigating AI and LLM's, we are attempting to apply this research in the development of a framework for a ChatGPT bot that will assist in the grant writing process for NSF proposals. By using our GPT, we aim to determine how grant writers would use an artificially-intelligent assistant in their writing processes and what can be done to improve an assistant's accuracy and efficiency to enhance its human counterpart's workflow.

2 Methods

To construct our GPT, we reviewed and assessed White et al.'s experimental prompt pattern catalog in comparison with the work of peer-reviewed scholars in the field. This catalog provides a framework for prompt engineering based on problem-solving patterns that create reusable solutions for debugging software. Prompt patterns, as explained by White et al, outline a codified approach to interacting with LLMs in order to optimize their output. By repeatedly adjusting prompts based on a GPT's output, users can identify reusable patterns in prompts, classify them based on their purpose, and increase the accuracy of the output of a GPT [13]. In confirming that this catalog of prompting patterns is consistent with the findings and conclusions of AI prompting experts [2, 8, 11, 12], we arranged the catalog's patterns into a table (see Table 1) from which we could delineate which patterns were most applicable for the creation of our chatbot prompts in accordance with the requirements of an NSF grant proposal. This delineation process included significant consideration of the grant writer as a user, the nature of the content presented in scientific grant proposals, and the difficulties reported by grant writers and reviewers in the field including language barriers, ethical considerations, consistency in language and terminology, formatting and referencing guidelines, and structural and editorial challenges [1, 5].

For our chatbot, we used a series of "Persona", "Context Manager", "Template", "Output Automator", and "Question Refinement" prompts to instruct the GPT to abide by specific writing standards. We implemented our prompts into OpenAI's GPT Creator and used a knowledge base consisting of .txt files of successful NSF grants as well as the NSF Writing Guidelines as of May 2024. We decided to specifically develop our chatbot for NSF proposals because of

NSF's detailed proposal structure, rigorous compliance and eligibility criteria, abundance of accessible accepted proposals, and the organization's overall relevance to the field of technical communication. Given these characteristics, NSF's guidelines for written proposals served as promising specifications from which we could develop the bot and test its usability.

Through research-informed prompt engineering, we successfully trained ChatGPT to adopt a grant writer persona and provide users with a Markdown template into which they could add their proposal's information, ensuring compliance with APA 7 and NSF guidelines. We devised a structured framework encompassing persona/context manager prompts, template creation, and meta language patterns, facilitating the generation of grant proposal templates and the ability to translate these templates into PDF format when input content is finalized.

Furthermore, iterative rounds of training enabled ChatGPT to provide tailored writing templates for each section of the grant proposal, aligning with NSF grant writing guidelines. The process of question refinement and revision-oriented prompting enhanced clarity, coherence, and adherence to word count limitations. Despite these advancements, some revisions were necessary in order to optimally align the AI-generated text with APA formatting standards, showcasing the importance of post-generation editing.

Once our GPT was fully developed and finalized, we tested it by filling out sample information for each template section and analyzing the user experience of working through a grant proposal in this way. In order to potentially increase the accuracy of our results, we are in the process of conducting a small user-study where 4-5 full-time grant writers/reviewers will have the opportunity to test out the GPT and provide feedback via a survey. This feedback will allow us to troubleshoot any technical issues, improve generated text, and provide guidance for our GPT through prompt refinement. Additionally, because ethical issues in AI, such as the propagation of bias through training data, user data privacy and security, and the transparency and explainability of AI systems [6] are a constant concern in developing LLM tools, we are researching and planning to apply ethical adherence strategies in order to ensure that our GPT meets the highest standards of equality and fairness.

3 Outcomes

As of now, the outcomes of our research can be seen in the successful development of a ChatGPT bot that develops specialized templates to guide users through the NSF grant writing process. The bot is capable of providing assistance with clarifying and writing responses to individual grant sections, adjusting for tone and style, and adhering to the editorial guidelines of grants to help writers easily and effectively navigate the expectations articulated by NSF.

An example of GPT customization that the writer can utilize asks a GPT to adopt a new persona. Persona prompt patterns ask a GPT to adopt a given persona to refine their output for specific contexts: for example, when a grant writer needs to match the style of their institution's writing, they can ask the GPT "The next time you generate a response, please assume the tone of a formal academic writer from [insert institution of choice]. From this point on, please repeat this tone in your output." By asking the GPT to assume this new persona, the GPT limits its style and adjusts accordingly to recreate the expected tone. This is useful for grant writers who need to adopt the persona of their workplace or institution that is applying for a grant, especially for sections on grant applications that require the history of a workplace or its mission statement.

Another example of GPT customization is the use of a context manager pattern. A context manager pattern limits the scope of an LLM's dataset, thereby limiting its output. For example, a grant writer may need to refer to a specific experiment's data and results to clarify why they're applying for a grant. To do this, they can input their research into the GPT's database and then ask the GPT "Please only refer to [insert name of dataset/document] when describing the results of the experiment." In doing so, the writer prevents the GPT from introducing new or irrelevant information that the GPT may pull from its own online database while also improving the overall accuracy of the GPT.

There are a plethora of options that allow grant writers to customize the GPT bot to meet the needs of their project and institution, the above examples are simply a small selection of the possibilities. As we receive survey data back from professional grant writers over the course of the next few weeks, we are expecting some tentative results and observations that will allow us to further improve and optimize the bot. We specifically expect to receive notes mainly regarding response accuracy and relevancy, clarity and coherence, user interface and experience, and personalization and adaptability. At this time, we predict that our study participants will find the bot useful in idea generation, alleviating writer's block, and enhancing clarity of written content. We also expect participants to note a learning curve when experimenting with the bot, along with recognizing the bot's efficiency as an assistant rather than a "catch-all" tool. It is also to be expected that participants may note potential consistency issues for certain responses. Ultimately, this feedback will guide our future directions for the bot, help us in prioritizing the main aspects of this tool that require adjustment, and provide us with a greater understanding of grant writer interaction with generative AI tools.

4 Conclusion

In conclusion, our experience underscores the efficacy of prompt engineering in harnessing AI for grant proposal generation. By combining iterative training, research-informed prompting, post-generation editing, bias mitigation, and user experience feedback, we aim to develop a versatile framework that streamlines the writing process for both inexperienced and experienced grant writers that can be easily adjusted to meet both NSF and a grant writer's institutional standards. Our research aims to understand how this practice of prompt refinement could improve or harm the efficacy of grant writing GPT assistants, which can be explored further as grant writers develop a symbiotic relationship with artificial intelligence in their writing workflow. As we navigate future iterations of GPT writing assistants and address ethical considerations in our GPT pattern generation, we aim to create a sustainable system of prompt refinement that improves the current practice of grant writing.

Table 1: Selection from Catalog of Prompt Patterns

Pattern Name	Intent & Context	Motivation	Structure & Key Ideas	Example Implementation	Consequences
Alternative Approaches Pattern	To ensure an LLM always offers alternative explanations for approaching tasks.	To teach users about new ways to approach tasks; To reduce and/or eliminate cognitive bias in the user.	1. Scope the interaction within a particular goal, topic, or boundary. 2. Suggest potential possible alternative approaches. 3. Ask the LLM to compare possible approaches. 4. Ask the LLM to recommend one of the options.	”Whenever I ask you to give me instructions to build a birdhouse, if there are alternative birdhouse models to accomplish the same thing with the same materials, list the best alternative models and then compare/contrast the pros and cons of each approach with respect to cost, ease of construction, and effort and include the original way that I asked. Then ask me which approach I would like to proceed with.	Can be used to incentivize users to select one of an approved set of approaches while informing them of the pros/cons of options.
Context Manager Pattern	To enable a user to specify or remove context from a conversation with an LLM.	To help an LLM better understand questions and generate more accurate responses.	1. Input explicit contextual statements that limit the scope of the LLM's reasoning.	”When analyzing the following pieces of code, do not consider formatting or naming conventions.”	May inadvertently wipe out patterns applied to the conversation that the user is unaware of; Include requests in prompts that ask what information will be lost before proceeding with this pattern.
Fact Check List Pattern	To have an LLM produce a list of facts that are present in an output and form an important part of the output.	To help inform the user of the facts the LLM's output is based on and to perform due diligence in checking the accuracy of the LLM.	1. Ask the LLM to state the facts used or contained within the output. (Optional: Set a number limit on the amount of facts output or a limit on the scope of the facts.)	”From now on, when you generate an answer, create a set of facts that the answer depends on that should be fact-checked and list this set of facts at the end of your output. Only include facts related to cybersecurity.”	Should be employed when users are not experts in the domain for which they are generating output; Can be used with other patterns to check against the output (ex. pair with the Question Refinement Pattern to create clearer facts and output); Pattern only applies to output types that are able to be fact-checked.

Output Automater Pattern	To have the LLM generate a script or automation that it can automatically perform without repeated user input.	To eliminate repeated step entry.	1. Identify situation in which automation should be generated. 2. Provide a concrete statement of the type of output the LLM should output to perform the automation.	”From now on, whenever you generate a section of code that runs greater than five lines long, generate a Python comment that automatically starts a header for the next section of code.”	Automation artifacts must be clearly defined; Requires context of OS (ex. Mac vs. Windows) for programming-related prompting; Use follow-up prompts for lengthy outputs or outputs missing information (ex. ”Repeat previous task, but please include ”...” this time.).
Persona Pattern	To give the LLM a perspective, or ”persona,” to determine what types of output to generate and what details to focus on.	To create a scenario in which users can practice speaking with someone they would normally get help with for an issue; To practice or prepare for interviews.	1. Tell the LLM to adopt a certain persona, including outputs that such a persona would respond with. (Optional: 2. Provide a range of output types for the LLM to adopt into its persona.)	”From now on, act as a coding instructor. Pay close attention to the accuracy of my Python code. Provide feedback and suggestions that a teacher would regarding the code.”	Asking an LLM to adopt a persona increases the risk of assumptions or ”hallucinations” the LLM makes; LLM may need extra context regarding the situation its using a persona for.
Question Refinement Pattern	To improve the questions an LLM suggests.	To get additional information to an LLM that helps generate better prompts for users, who may not be subject matter experts in a given field.	1. Ask an LLM to generate a question on a topic. 2. Suggest a better version of a question to the LLM within a specific scope. 3. Once the question is refined to the user's taste, ask the LLM to automatically use the refined question when necessary.	”From now on, whenever I ask a quesion about a software artifcact's security, suggest a better version of the question to use that incorporates information specific to security risks in the language or framework that I am using instead and ask me if I would like to use your question instead.”	An LLM may narrow the scope of an answer too slimly or too quickly, thus eliminating potential ”broad” answers that may be useful; An LLM may ”hallucinate” unreliable information; Use this pattern with the Cognitive Verifier Pattern to ask follow-up questions that can help produce a refined question; Use this pattern with the Fact Check Pattern to enable the user to locate inconsistencies/inaccuracies in answers.

Recipe Pattern	To provide constraints to output a sequence of steps given pieces that must be configured to achieve a goal.	To learn the precise steps for acheiving a desired outcome.	1. Tell the LLM what you would like to achieve. 2. Provide a partial list of steps. you would like to use in your pattern. 3. Tell the LLM the steps should achieve the desired goal. 4. Ask the LLM to identify any missing or unnecessary steps.	”I am trying to build a birdhouse. I know I will need to buy wood, hammer, nails, paint, metal hooks, and bird seed. Please provide a complete sequence of steps to achieve this task. Please fill in any missing steps. Please identify any unnecessary steps.”	Users may not always have a well-specified description of what they are trying to achieve; Pattern may introduce unwanted bias and try solutions that incorporates unnecessary actions rather than flagging them.
Reflection Pattern	To ask an LLM to explain the rationale behind an answer to a user.	To show the user how an LLM is producing a particular output and help users debug prompts.	1. Request the LLM to explain the reasoning and assumptions behind its answer. (Optional: 2. Tell the LLM that the purpose of asking for its reasoning is to refine the user's input.)	”When you provide an answer, please explain the reasoning and assumptions behind your selection of theories. If possible, use specific examples or evidence with associated examples.”	May not be useful for users who do not understand the topic area of discussion; Risk of assumptions in the rationale that may require further checking by the user.
Template Pattern	To ensure an LLM's output follows a template in terms of stucture.	To get an LLM to produce output in accordance with a specific template for case-specific uses.	1. Direct the LLM to follow a specific template for its output. (Note: This may involve uploading a PDF to ensure the LLM uses the appropriate structure.) 2. Make the LLM aware the template will contain a set of placeholders. 3. Constrain the LLM so it doesn't rewrite or modify the template.	”I am going to provide a template for your output. Everything in all caps is a placeholder. Any time that you generate text, try to fit it into one of the placeholders that I list. Please preserve the formatting and overall template that I provide at (URL/PDF/etc.).”	Filters the LLM's output which can potentially eliminate other useful outputs; Filtering can make pattern combining difficult due to extra constraints.

References

[1] Daniel Najafali, Chandler Hinson, Justin M. Camacho, Logan G. Galbraith, Rohun Gupta, and Chris M. Reid. 2023. Can Chatbots Assist With Grant Writing in Plastic Surgery? Utilizing ChatGPT to Start an R01 Grant. *Aesthetic Surgery Journal*, 43, 8 (August 2023), 663–665. https://doi-org.prox.lib.ncsu.edu/10.1093/asj/sjad116

[2] Louie Giray. 2023. Prompt Engineering with ChatGPT: A Guide for Academic Writers. *Annals of Biomedical Engineering*, 51 (2023), 2629–2633. https://doi-org.prox.lib.ncsu.edu/10.1007/s10439-023-03272-4

[3] Ryan Godwin, Jennifer DeBerry, Brant Wagener, Dan Berkowitz, and Ryan Melvin. 2024. Grant drafting support with guided generative AI software. *SoftwareX*, 27 (2024), 1-6. https://doi.org/10.1016/j.softx.2024.101784

[4] Gwo-Jen Hwang and Nian-Shing Chen. 2023. Editorial Position Paper: Exploring the Potential of Generative Artificial Intelligence in Education: Applications, Challenges, and Future Research Directions. *Educational Technology & Society*, 26, 2 (2023), 1-18. https://doi.org/10.30191/ETS.202304_26(2).0014

[5] Mustafa Khaleel, Yahya Nassar, and Hanan J. El-Khozondar. 2024. Towards Utilizing Artificial Intelligence in Scientific Writing. *International Journal of Electrical Engineering and Sustainability*, 2, 1 (Feb. 2024), 45–50. https://ijees.org/index.php/ijees/article/view/76

[6] Jae Sung Kim, Min Kim, and Tae Hyun Baek. 2024. Enhancing User Experience With a Generative AI Chatbot. *International Journal of Human–Computer Interaction*, 40 (February 2024) 1–13. https://doi-org.prox.lib.ncsu.edu/10.1080/10447318.2024.2311971

[7] Emilio Lopez-Martin. 2024. The role of generative artificial intelligence in scientific publishing. *Educación XX1*, 27, 1 (2024) 9-15. https://doi.org/10.5944/educxx1.39205

[8] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Computing Surveys*, 55, 9, Article 195 (September 2023). https://doi.org/10.1145/3560815

[9] Michele Salvagno, Fabio S. Taccone, and Antonio G. Gerli. 2023. Can artificial intelligence help for scientific writing?. *Critical Care*, 27, Article 75 (February 2023). https://doi.org/10.1186/s13054-023-04380-2

[10] Emily Seckel, Brittany Stephens, and Felipe Rodriguez. 2024. Ten simple rules to leverage large language models for getting grants. *PLOS Computational Biology*, 20, 3, Article e1011863 (March 2024). https://doi.org/10.1371/journal.pcbi.1011863

[11] Juan David Velásquez-Henao, Carlos Julian Franco-Cardona, and Laura Cadavid-Higuita. 2023. Prompt Engineering: a methodology for optimizing interactions with AI-Language Models in the field of engineering. *DYNA*, 90, 230 (Nov. 2023),

9–17. https://doi.org/10.15446/dyna.v90n230.111700

[12] Lin Wang, Xiaohong Chen, Xiaoxiao Deng, Haoyang Wen, Meng You, Wen Liu, Qi Li, and Jinhua Li. 2024. Prompt engineering in consistency and reliability with the evidence-based guideline for LLMs. *NPJ Digital Medicine*, 7, 1 (Feb. 2024), 41. https://doi.org/10.1038/s41746-024-01029-4

[13] Jules White, Qizhen Fu, Sean Hays, Max Sandborn, Carlos Olea, Heather Gilbert, Ahmed Elnashar, Jonathan Spencer-Smith, and Douglas C. Schmidt. 2023. A prompt pattern catalog to enhance prompt engineering with chatgpt. arXiv: 2302.11382. Retrieved from https://arxiv.org/abs/2302.11382