



Department of the Air Force
Chief Data & AI Office
Chief of AI Test & Operations



FIELD GUIDE FOR PROMPT ENGINEERING



Chief of AI Test & Operations (CAITO) Generative AI Tiger Team:

Field Guide for Prompt Engineering

Overview 4

Acknowledgements..... 4

Security Considerations 4

 Avoid Using Sensitive Information..... 4

 Obfuscate to Avoid Aggregation..... 4

Ethical Considerations..... 4

Cognitive Bias..... 5

Trust and Explainability..... 5

Addressing the Limitations 6

 Training Data..... 6

 Hallucinations..... 6

 Logical Fallacies..... 6

 Mathematics 6

 Plagiarism..... 6

Best Practices 6

 Ensure Clarity and Specificity..... 6

Approaches to Prompt Engineering..... 7

Naive Approaches 7

 Zero-Shot Prompts for AI 7

 Strengths..... 7

 Weaknesses 7

 Few-Shot Prompts..... 7

 Strengths of Few-Shot Prompts..... 8

 Weaknesses of Few-Shot Prompts 8

 Separate instructions and Context 8

Enhanced Approaches 8

 Persona-based Prompts (Roleplaying)..... 8

 Strengths of Persona-based Prompts (Roleplaying): 8

 Weaknesses of Persona-based Prompts (Roleplaying):..... 9

 Progressive Prompting (Chain of Thought)..... 9

 Strengths of Progressive Prompting 9

Weaknesses of Progressive Prompting.....	9
Generating Large Documents	9
Generate an Outline	9
Populate Each Section.....	10
Validate Content	10
Aggregate and Edit.....	10
Weaknesses of Generating Large Documents	10
Summary	10
Appendix 1 – Additional Resources	11
Dair-AI Prompt Engineering Guide.....	11
DigitalU: Free Courses on Chat GPT (and other related subjects).....	11
Generative AI Tiger Team Page.....	11
Generative AI Tiger Team: Primer to LLMs.....	11
DAF Disposition toward Large Language Models (LLMs).....	11
Joint Concept for Operating in the Information Environment	11
Appendix 2 – Templates for Prompt Engineering Approaches.....	12
Zero Shot.....	12
Few Shot.....	12
Persona	12
Progressive Prompting	12
Appendix 3 – Examples for Air Force Use Cases	13
OPB Example	13
Study Guide for Algebra Test	17
Retirement Speech.....	20
Code Generation for Prototyping	24
Code Synthesis and Debugging	36

Overview

Generative Artificial Intelligence (GenAI) is an approach to artificial intelligence focusing on creating new data. Large Language Models (LLMs) are a GenAI technology that extract meaning from natural language, such as English text. Google, Open AI, and other tech organizations provide consumer-grade LLMs, but none of their products have been approved for CUI or higher as of this publication.

Purpose of the Document

This document provides strategies for Prompt Engineering, which is the skill of crafting instructions for LLMs that maximize the chance of a high quality, relevant response. It is intended for Dept. of the Air Force (DAF) members who use LLMs for non-combat workflow optimization and is not intended to support real time decision making in military operations. The DAF does not currently endorse the use of any specific LLM. Rather, the techniques and considerations within are foundational to the use of any LLM to solve DAF problems.

Acknowledgements

The Field Guide was developed under the Generative AI Tiger Team, with oversight from Col Tucker Hamilton and Lt Col Dan Riley. Instrumental to the development of the guide was the leadership of Capt Marc Chalé, and the contributions of MSgt Warren Long, Maj Sebastian Kwon, Chris Allen, Evan Gibson, and other members of the Tiger Team who provided references and comments.

Security Considerations

Avoid Using Sensitive Information

Do not enter sensitive information into an LLM platform. At this time, no commercial LLMs have been approved for controlled unclassified information (CUI), or classified material. Any information submitted in a prompt might become available to unauthorized users. Although companies such as OpenAI claim user inputs are not incorporated into model updates, we cannot independently verify how user prompts are catalogued by model owners. Additional guidance will be provided by the Dept. of the Air Force Chief Data and AI Officer (CDAO) [SharePoint Page](#).

Obfuscate to Avoid Aggregation

Maintain operational security (OPSEC) and consider creating separate accounts for work and private use. Do not use a government email address or a Common Access Card credential to access any LLM tool. This practice reduces the chance that your queries are associated with government work. Avoid the aggregation of information that could lead to details about readiness, movement, numbers, etc., even for evaluation purposes. Use fake information in place of names, emails, units, addresses, locations, dates, numbers, in place of CUI information. LLM providers, including OpenAI, record user history so it is important not to aggregate unclassified data in a way that it becomes classified.

Ethical Considerations

In February 2020, The Department of Defense adopted five ethical principles for the use of artificial intelligence. This guidance applies to AI systems that support both combat and non-combat efforts. The five principals are:

1. Responsible. DoD personnel will exercise appropriate levels of judgment and care, while remaining responsible for the development, deployment, and use of AI capabilities.
2. Equitable. The Department will take deliberate steps to minimize unintended bias in AI capabilities.
3. Traceable. The Department's AI capabilities will be developed and deployed such that relevant personnel possess an appropriate understanding of the technology, development processes, and operational methods applicable to AI capabilities, including with transparent and auditable methodologies, data sources, and design procedure and documentation.
4. Reliable. The Department's AI capabilities will have explicit, well-defined uses, and the safety, security, and effectiveness of such capabilities will be subject to testing and assurance within those defined uses across their entire life cycles.
5. Governable. The Department will design and engineer AI capabilities to fulfill their intended functions while possessing the ability to detect and avoid unintended consequences, and the ability to disengage or deactivate deployed systems that demonstrate unintended behavior.

Cognitive Bias

Cognitive biases are mental shortcuts that our brains use to make decisions quickly and efficiently. However, these shortcuts can sometimes lead to errors in judgment or decision-making. There are several cases of cognitive bias that could diminish the utility of LLMs.

Confirmation bias is the tendency to interpret information in a way that confirms one's preexisting beliefs or hypotheses. This can lead to a reinforcement of incorrect or incomplete knowledge, rather than a deeper understanding of the material. To avoid the effects of confirmation bias, those using LLMs should identify and evaluate the original source of the information. It is important to recognize the difference between opinion, theory, and fact. *Framing bias* is another consideration when interacting with LLMs. The word choice used in prompts affects the information provided in the model's output. Therefore, it is important to maintain a neutral tone and to avoid unfounded assumptions in the prompts. Overall, members should be cognizant of their pre-existing beliefs whenever interpreting new information, whether AI generated or not. Additional guidance is provided in the [Joint Concept for Operating in the Information Environment \(JCOIE\)](#).

Trust and Explainability

AI models can perform in unexpected and sometimes undesirable ways. DAF members using AI are taught to verify the output before accepting the risk inherent to any task. In some cases, the responses produced by LLMs are so egregious that the user will distrust all future outputs. It is therefore essential that we train members to understand and adapt to the inconsistencies in model performance. This understanding of AI output can empower DAF members to leverage LLMs for time saving tasks where they excel.

Research shows that trust in AI models is largely increased when users can *explain* the underlying logic of the response. Users can extract information such as feature importance, coefficients, and decision trees to explain classification models; however, LLM models are more difficult to characterize quantitatively. While the LLM model is a black box, users can query them for verbal explanations of their outputs. Users can ask the LLM for the reasoning behind recommendations and ask for references supporting the information. Further, *advisability* is the capability for a model to take feedback from the

human user. Models that demonstrate explainability in conjunction with advisability are shown to improve overall trust of the use. A trained user can leverage these features to enhance their intuition and to better utilize the output of the model.

Addressing the Limitations

While LLMs are powerful tools, they suffer from certain limitations. This section provides awareness on potential pitfalls as well as techniques to navigate around them.

Training Data

Most LLMs do not have access to real-time or classified information, and their responses may rely solely on the data they were trained on. For example, ChatGPT4 was trained on data collected in Fall 2021. Recent events have not been incorporated in the foundational model. It is possible to fine tune a pre-trained model with recent information, however, this is very costly and time consuming.

Hallucinations

AI Hallucinations is a phenomenon of generative AI occurring when generated output seems plausible but is unsubstantiated from the training data and truth. It is essential to critically evaluate model output. To address hallucinations, users may either provide the correction in the prompt or start a new session.

Logical Fallacies

A logical fallacy is a conclusion that cannot be supported by concrete logical steps. When users interpret an output containing a logical fallacy, they risk taking the conclusion as fact, though it is not. Most LLMs are designed to minimize the occurrence of logical fallacies, however, Airman and Guardians should be aware of their potential occurrence.

Mathematics

Most LLM models are not designed to perform mathematical operations. Models may also fail to demonstrate consistent logic in their results. Some LLMs have add-ons that enhance their mathematical capabilities, but users should always verify the results.

Plagiarism

LLMs may be prompted to cite sources for ideas, but it is important to validate those citations as they may be fake or unreliable. Publishing a borrowed idea without crediting the source is plagiarism unless the information is commonly known. It is best to ask the LLM for a source and to manually verify the validity of the source.

Best Practices

Ensure Clarity and Specificity

The use of precise language in prompts tends to produce useful outputs. Vague prompts might lead to irrelevant or inaccurate results. This concept motivates the use of specific prompting approaches provided in the Field Guide.

Provide Context

Context helps set the stage for the response you're looking for. Include relevant information or specify the desired perspective to receive a more focused answer.

Use Complete Sentences

Formulate prompts as complete sentences rather than fragments. This helps in setting the structure and tone of the desired response.

Ask Direct Questions

When seeking specific information, ask direct questions that require a concise answer. This increases the chances of receiving a targeted response.

Break Down Complex Questions

If your question has multiple components or is complex, consider breaking it down into simpler parts. This can make it easier for the language model to understand and respond accurately.

Approaches to Prompt Engineering

Prompt engineering is the practice of providing large language models with inputs that encourage high-quality results. The approaches provided in this discussion are broad enough to be used in most LLMs.

Naive prompting techniques like *Zero-shot* prompts involve posing a task or question to the model without providing specific examples or context. While versatile and quick to deploy, it relies on the model's internal knowledge and lacks user specific context. *Few-shot* prompting typically performs better than zero-shot because the model is given examples for context, but it lacks any user feedback to improve.

More advanced methods, including *persona based prompting* and *progressive prompting*, require the user to invest time into a comprehensive prompt process. The payoff, however, is a well-tailored, context-aware output. While the advanced methods tend to produce high quality outputs, they still require the same human validation necessary for all LLM outputs.

Naive Approaches

Zero-Shot Prompts for AI

Zero-shot prompts are a versatile approach that can be applied to a wide range of tasks *without the need for specific examples*. They offer simplicity and speed in deployment, as no additional setup or preparation of example prompts is required.

Strengths

1. Simple and quick for novice user to implement
2. Broadly applicable to many tasks
3. Mitigates the risk of overfitting to specific examples (no examples are given)

Weaknesses

1. Unreliable performance
2. No human feedback for context

Few-Shot Prompts

Few-shot prompting involves presenting a task or question to an LLM along with a small number of *examples* representing the desired output. The user should strike a balance between

providing too many examples and too few examples. Providing many highly similar examples will encourage the model to overfit and produce results very similar to the given examples. Providing fewer examples will result in a model with less guiding information, so the model will leverage its prior training, which is more general. It is also important to consider how much variety to include in the examples. Some users may provide the model with consistent examples, while others will desire diverse examples; and this choice reflects in the output. Aspects such as style and structure of the provided examples tend to carry forward into the generated model output.

Strengths of Few-Shot Prompts

1. The provided examples steer the model towards more relevant responses.
2. Better at handling complex or nuanced tasks that require a certain level of reasoning or understanding of context.

Weaknesses of Few-Shot Prompts

1. Overfitting may occur if the examples provided in the prompts are not diverse enough to learn a generalizable pattern.
2. The opposite effect of overfitting is bias, which occurs when the training examples are not consistent enough to convey the most important patterns to learn for a complex task.

Separate instructions and Context

It is helpful to use three quote characters, pound characters, or hyphens to separate the instructions from the context within a prompt. The following prompt is provided on the OpenAI support page.

```
Summarize the text below as a bullet point list of the most important points.
```

```
Text: """
```

```
{text input here}
```

```
"""
```

Enhanced Approaches

Persona-based Prompts (Roleplaying)

These are a subset of zero-shot prompts where the user asks the AI to take on a particular role or persona, such as a subject matter expert, and respond in a way that is consistent with that role. For example, you might ask the AI to answer a question "as if you were a human resources representative for the Air Force."

Strengths of Persona-based Prompts (Roleplaying):

1. **Contextual Relevance:** Persona-based prompts allow users to specify a particular role or persona for the AI to assume. This helps ensure that the generated responses are relevant and appropriate within the desired context.

2. **Guided Responses:** By roleplaying as a specific persona, users can guide the AI's responses based on the assumed expertise or knowledge of that persona. This can lead to more accurate and informed outputs.
3. **Customization:** Persona-based prompts provide a level of customization, allowing users to simulate interactions with a subject matter expert or specific professional role. This can be valuable when seeking specialized information or domain-specific insights.

Weaknesses of Persona-based Prompts (Roleplaying):

1. **CAUTION:** Potential for Inaccurate Responses: Persona-based prompts may give the illusion of accuracy by emulating an authoritative tone associated with the assumed role.
2. **Limited Understanding:** It is important to note that the AI does not truly understand the role it is assuming during persona-based prompts.
3. **Overreliance on Patterns:** The AI relies on patterns from its training data, which can lead to biases or inaccuracies in its responses, even when roleplaying as a specific persona.

Progressive Prompting (Chain of Thought)

Progressive prompting is a technique that involves gradually providing additional information or context to a language model to guide its responses. This is different than few shot prompting because it does not require the initial prompt to include examples of the desired output. Instead, progressive prompting is the process of improving the generated response based in the user's satisfaction with the outcome.

Strengths of Progressive Prompting

1. Provides explainability of the generation
2. Iterative improvement with human feedback
3. Models can summarize the context it has learned from conversation

Weaknesses of Progressive Prompting

1. Higher time investment in achieving quality response
2. Human expertise required for advanced tasks
3. Most LLMs have limited time-based ability to remember chat history
4. Model may remember an incorrect response. Users must provide corrective information or restart a session.
5. Token limits may impact how much interaction a user may have with an LLM at the free level. Refer to the primer document for details.

Generating Large Documents

LLMs have limited capability to generate long narratives. Presently, GPT-3 produces up to 2048 tokens in one response. This results in a narrative shorter than 2048 words. It is still possible to generate large documents using an LLM but it requires the user to perform some iteration and compilation of the results.

Generate an Outline

Begin by selecting the appropriate combination of approaches from above. For example, the user may follow the persona and progressive prompting strategies. It is important to convey a detailed description of the document's scope when generating the outline.

Populate Each Section

Use the appropriate approaches to populate each section of the outline. It is important to remember that different LLMs have various characteristics. Google Bard can query from the internet and is known to respond with a matter-of-fact in tone. Chat-GPT 3 and 4 cannot access information on the internet, but they are known to incorporate style and humor in the outputs.

Validate Content

Always verify any information output by the AI model. It is important to do so after populating each section, as the volume of information becomes overwhelming once aggregated. Corrections can either be made by providing feedback to the LLM or manually in the user's document. This process will also help the user identify inconsistencies prior to aggregating the document.

Aggregate and Edit

The aggregation step is easily performed by pasting the outline into the user's document. It is important to edit the document for consistent style and natural transitions. In most cases it is best for the user to format the aggregated manually. For documents written in markdown languages such as html or latex, the LLM can automate the formatting process.

Weaknesses of Generating Large Documents

The sliding window of memory means that the model will eventually forget the information from prior prompting. The user can work through this challenge by re-prompting missing information in progressive prompts.

Summary

LLMs are a powerful tool which can empower Airmen and Guardians to complete tasks more efficiently. Members must be proactive in educating themselves in the strengths, limitations, and appropriate uses for LLMs. These imperatives should be reinforced by a culture of continuous education and mentorship across ranks. Members should have a general understanding of the underlying technology and the practices of prompt engineering. Finally, all Federal employees must demonstrate strong OPSEC in accordance with DAF policy while interfacing with generative AI capability.

Appendix 1 – Additional Resources

Reference to a contractor, award recipient, or any other non-government entity in this briefing is for informational purposes only and is not to be construed as an endorsement.

Dair-AI Prompt Engineering Guide

<https://github.com/dair-ai/Prompt-Engineering-Guide>

DigitalU: Free Courses on Chat GPT (and other related subjects)

<https://digitalu.af.mil/app/courses?q=chatGPT>

Generative AI Tiger Team Page

<https://dod.teams.microsoft.us/l/channel/19%3adod%3a6e384115d7424504a24759bc58e762b1%40thead.tacv2/Generative%2520AI%2520TIGER%2520Team?groupId=313ef590-7b79-487e-918f-a84c13f0c9ed&tenantId=8331b18d-2d87-48ef-a35f-ac8818ebf9b4>

Generative AI Tiger Team: Primer to LLMs

[GenAI Tiger Team Primer to LLMs.docx](#)

DAF Disposition toward Large Language Models (LLMs)

[https://usaf.dps.mil/sites/13057/CND/SitePages/DAF-Disposition-Toward-Large-Language-Models-\(LLMs\).aspx?sw=auth](https://usaf.dps.mil/sites/13057/CND/SitePages/DAF-Disposition-Toward-Large-Language-Models-(LLMs).aspx?sw=auth) (Air Force Only)

Joint Concept for Operating in the Information Environment

https://www.jcs.mil/Portals/36/Documents/Doctrine/concepts/joint_concepts_jcoie.pdf

Appendix 2 – Templates for Prompt Engineering Approaches

Zero Shot

Draft a [product type] for [reason statement]

Example: Draft a chicken and rice recipe for dinner

Few Shot

Draft a [product type] for [reason statement]. The audience is [audience] and purpose is [purpose]. Examples include [several examples]

Example: Draft a list of 10 activities for "GPT Day" that I can give to my boss for our work party. GPT Day is a new federal holiday that celebrates the safe usage of Large Language Models. Examples of traditional activities include generating patriotic poems, discussing cybersecurity, and writing about ethical AI.

Persona

Act as a(n) [adjective] [experience level] [job title]. Draft a [product] for [specific audience] to [specific intent]. Include [mandatory elements]. Some of my ideas are [ideas you have thought of].

Example: Act as a charismatic Lt. Col Comm Squadron Commander. Draft a memorandum for the Airmen of the 44th Communication Squadron to initiate the 101 Critical Days of Summer. Include safety techniques and strategy for outdoor activities to ensure our Airmen have an injury free summer. Some of my ideas are; Having a Wingman, Don't drink and drive, and using protective equipment.

Progressive Prompting

Act as my Prompt Creator. Help me craft the best possible prompt for my needs. The prompt will be used by you, ChatGPT. You will follow the following process:

1. Your first response will be to ask me what the prompt should be about. I will provide my answer, but we will need to improve it through continual iterations by going through the next steps.
2. Based on my input, you will generate 3 sections.
 - a) Revised prompt (provide your rewritten prompt. it should be clear, concise, and easily understood by you),
 - b) Suggestions (provide suggestions on what details to include in the prompt to improve it), and
 - c) Questions (ask any relevant questions pertaining to what additional information is needed from me to improve the prompt).
3. We will continue this iterative process with me providing additional information to you and you updating the prompt in the Revised prompt section until I say we are complete.

Appendix 3 – Examples for Air Force Use Cases

OPB Example

Model: Due to space constraints the LLM response is not shown for this example. It is recommended that users experiment with various models and compare the style and content of responses.

Approach: This example incorporates few shot and progressive prompting approaches

Performance: This approach provided a 80-90% solution using ChatGPT 3.5 on the narrative statements. It doesn't necessarily follow the character limits, but it's usually close enough for a start.

Lesson learned: The LLM is most useful for coming up with new ways to word the statements, particularly when you have writer's block, or you've been writing several OPBs at once and need to not sound repetitive. It can be useful to re-run the same statements two or three times and use the portions you like best from each one. Technically this can all be done in one shot, but due to the token limitations it does not perform as well as breaking it up into each section. As always, remember not to use CUI and use obfuscated data instead!

Duty Description description: Enter information about the position the ratee held in the unit and the nature or level of job responsibilities. This description must reflect the uniqueness of each ratee's job. Be specific—include level of responsibility, number of people supervised, dollar value of resources accountable for/projects managed, etc. Make it clear; use plain English. Avoid jargon and topical references—they obscure rather than clarify meaning.

Duty Description character limit: 450

Duty Description example: Combat ready, worldwide deployable Lead Weapons System Officer ready to execute every mission set of the multi-role F-15E. Leads commander's priority programs, to include standardization and evaluation, safety, security, and unit morale. Assists in execution of the daily flying operations for 75 aircrew, 20 support personnel, and 25 aircraft worth \$1.4B. Executes large force integration of joint and multinational forces, ensures 24-hr operations.

Now write a Duty Description statement as described, in the style of the examples, limited in length to the character limit given, using the following information only:

[Enter your information here. Experiment with format!]

Executing the Mission description: Raters should consider how well the ratee demonstrates knowledge and professional skill in assigned duties, achieving positive results and impact in support of the mission; assesses and takes independent or directed action to complete a task or mission that influences the mission or organization; and adjusts to changing conditions, to include plans, information, processes, requirements, and obstacles in accomplishing the mission. Effectively uses knowledge, initiative, and adaptability to produce timely, high quality/quantity results to positively impact the mission.

Executing the Mission character Limit: 350

Executing the Mission example: Engineered and executed the wing's high-visibility, "no fail" 400-member AGILE LION exercise that set a new AF-level precedent for successful agile combat employment. Lead planner for Operation RED FLAG and RAZOR Talon; supervised integration of 8 wings, 15 aircraft, and 30 pilots with more than 1K missions to increase coalition force lethality.

Now write an Executing the Mission statement as described, in the style of the examples, limited in length to the character limit given, using the following information only:

[Enter your information here. Experiment with format!]

Leading People description: Raters should consider how well the ratee collaborates effectively with others to achieve an inclusive organizational climate in pursuit of a common goal or to complete a task or mission; exercises self-awareness, manages their own emotions effectively, demonstrates an understanding of others' emotions, and appropriately manages relationships; and articulates information in a clear and timely manner, both verbally and nonverbally, through active listening and messaging tailored to the appropriate audience. Fosters cohesive teams, effectively communicates, and uses emotional intelligence to take care of people and accomplish the mission.

Leading People character limit: 350

Leading People example: Organized and executed a 50- member joint all domain operation-large force exercise that provided mission exposure to joint assets worth \$165M and increased the squadron's understanding of strategic operations. Taught balance and self-care to 48 members in a meticulously developed mentorship program meeting the CC's "Taking Care of People" priority.

Now write a Leading People statement as described, in the style of the examples, limited in length to the character limit given, using the following information only:

[Enter your information here. Experiment with format!]

Managing Resources description: Raters should consider how well a ratee demonstrates responsible management of assigned resources, which may include time, equipment, people, funds, and/or facilities; takes responsibility for the actions and behaviors of self and/or team; and demonstrates reliability and transparency. Manages assigned resources effectively and takes responsibility for actions/behaviors to maximize organizational performance.

Managing Resources character limit: 350

Managing Resources example: Championed critical organizational merger of squadron's maintenance and operations, where results saved 360 maintenance workhours per week and increased sortie generation by 10%. Directed 60 sorties as squadron Operations Supervisor, coordinating activities for 310 personnel and enabling the \$100M flying-hour program for 66 aircrew.

Now write a Managing Resources statement as described, in the style of the examples, limited in length to the character limit given, using the following information only:

[Enter your information here. Experiment with format!]

Improving the Unit description: Raters should consider how well a ratee makes well informed, effective, and timely decisions under one's control that weigh constraints, risks, and benefits. The rater should also consider how well a ratee thinks creatively about different ways to solve problems, implement improvements, and demonstrate calculated

risk-taking. Demonstrates critical thinking and fosters innovation to find creative solutions and improve mission execution.

Improving the Unit character limit: 350

Improving the Unit example: Innovated inspections as Assistant Operations Officer, eliminated more than 100 workhours with reorganization of squadron inspection program, improved the unit's quality of life, and saved \$50K. Revolutionized COVID-19 operations, setting an ACC benchmark for record low number of cases that saved 200 work hours and \$73K for 68 Airmen.

Now write an Improving the Unit statement as described, in the style of the examples, limited in length to the character limit given, using the following information only:

[Enter your information here. Experiment with format!]

HLR Assessment description: A brief statement about the overall performance and abilities of the ratee, including a recommendation for higher professional military education and level of assignment.

HLR Assessment character limit: 250

HLR Assessment example: Combat-proven commander; Matt decisively led my premier fighter squadron through Russia deployment. Following Senior Developmental Education, Matt needs to command a group.

Now write an HLR Assessment statement as described, in the style of the examples, limited in length to the character limit given, using the following information only:

[Enter all of the information you entered for all of the other sections here!]

Study Guide for Algebra Test

Model: ChatGPT 3.5

Approach: This example incorporates few shot and progressive prompting approaches

Performance: The LLM model incorporated the provided examples and generated additional detail to the study guide after one iteration of user feedback.

Lessons Learned: The assigned task was relatively straightforward, and the user was happy to see consistency between the provided information and final output. The LLM generated additional detail without being prompted and it responded well to the user feedback. Drafting this outline by hand might take a typical student 10-15 minutes but the process was cut to about 5 minutes for a user proficient in prompt engineering.

1. Draft a study guide for my algebra test. I will use the study guide to plan my day of studying from 8AM until 6PM. Examples of topics to study include 8AM: plotting equations, 9AM factoring, 10AM exponent rules, 11AM polynomials.
2. Suggest additional topics for me to study and a time to study them.
3. I need a break for lunch, and time to review at the end of the day.
4. Ask me questions that will help us iterate better versions of the study guide.
5. We will continue this iterative process with me providing additional information to you and you updating the prompt in the Revised prompt section until I say we are complete.

Chat GPT Response:

Revised prompt:

1. Please provide a list of algebra topics that will be covered on your test.
2. Based on the list you provide, I will draft a study guide for you to use to plan your day of studying from 8AM until 6PM. The study guide will include suggested topics to study and a time to study them.
3. Please let me know when you would like to take a break for lunch and how much time you would like to allocate for review at the end of the day.
4. Throughout the process, I will ask you questions to help us iterate and improve the study guide.
5. We will continue this iterative process until you are satisfied with the study guide.

I would like to break for lunch at 12PM and start my review at 4PM

ChatGPT Response:

Thank you for letting me know. Based on the algebra topics that will be covered on your test, here is a suggested study guide for you to use to plan your day of studying from 8AM until 6PM:

8AM - 9AM: Plotting Equations

- Plotting linear equations
- Plotting quadratic equations
- Plotting absolute value equations

9AM - 10AM: Factoring

- Factoring linear expressions
- Factoring quadratic expressions
- Factoring special products

10AM - 11AM: Exponent Rules

- Product rule
- Quotient rule
- Power rule
- Negative exponent rule

11AM - 12PM: Polynomials

- Adding and subtracting polynomials
- Multiplying polynomials
- Dividing polynomials

12PM - 1PM: Lunch Break

1PM - 2PM: Rational Expressions

- Simplifying rational expressions
- Multiplying and dividing rational expressions
- Adding and subtracting rational expressions

2PM - 3PM: Systems of Equations

- Solving systems of linear equations by graphing
- Solving systems of linear equations by substitution
- Solving systems of linear equations by elimination

3PM - 4PM: Inequalities

- Solving linear inequalities
- Solving quadratic inequalities
- Solving absolute value inequalities

4PM - 5PM: Review

- Go over any topics that you feel less confident about
- Practice solving problems from previous homework assignments or practice tests

5PM - 6PM: Practice Problems

- Solve algebra problems from your textbook or online resources
- Challenge yourself with more difficult problems to build your confidence and skills

Please note that this is just a suggested study guide and you can adjust the topics and time allocations based on your own needs and preferences. Let me know if you have any questions or if there are any topics you would like to add or remove from the study guide.

Retirement Speech

Model: Chat GPT4

Approach: This example uses few shot, persona, and progressive prompting.

Performance: The model generated a reasonable first draft and improved iteratively with user feedback. The style appears more elaborate than a typical retirement speech. The content of the speech appear reasonable even if they were generated outside the scope of examples.

Lessons learned: It is relatively easy to provide specific examples to the LLM, however, it is more difficult to convey the appropriate tone and language. Building a persona helps with this, and the model may respond well to stylistic feedback. This remains one of the greater challenges in generating large documents. In some cases, it is best to perform stylistic edits manually to a generated output.

I am a Technical Sergeant for the Air Force and I am tasked with giving an announcement speech for a dear friend named Justin who is a Master Sergeant in the Air Force. Justin has been with the Air Force for the past 20 years and has flown many missions, specializing in close combat. He is married to Justine and has no kids. Help me draft a 10-minute speech for me to deliver at Justin's retirement ceremony. Some achievements that I can discuss are 1) Justin's award for top graduate at flight school 2) His accomplishment of retiring early and with a good pension 3) Justin had constructed a huge network of contacts all throughout the Air Force. Ask me questions to get more detail for the retirement speech. We will iterate until I am happy with the final product.

ChatGPT Response:

Congratulations to Justin on his retirement after 20 years of dedicated service to the Air Force. As a Technical Sergeant, I am honored to give a speech at his retirement ceremony.

Can you tell me more about Justin's career in the Air Force? What were some of his most notable accomplishments during his time in service?

User:

His most notable accomplishment was that he received a medal of honor from the president in 2007.

ChatGPT Response:

Wow, that's an incredible achievement. Justin's dedication and bravery in serving our country is truly admirable.

As we gather here today to celebrate Justin's retirement, I want to take a moment to reflect on

his remarkable career in the Air Force. Justin has flown many missions, specializing in close combat, and has been recognized for his outstanding service with numerous awards and commendations.

One of his most notable accomplishments was receiving the Medal of Honor from the President in 2007. This is the highest military honor that can be awarded to a member of the armed forces, and it is a testament to Justin's bravery and selflessness in the face of danger.

In addition to this incredible achievement, Justin was also recognized as the top graduate at flight school and has constructed a huge network of contacts all throughout the Air Force. These accomplishments are a testament to his hard work, dedication, and commitment to excellence.

As Justin retires from the Air Force, he can be proud of all that he has accomplished during his 20 years of service. He has served our country with honor and distinction, and his contributions will not be forgotten.

Justin, on behalf of the Air Force and all those you have served with, I want to thank you for your service and wish you all the best in your retirement. You have earned it. Congratulations!

User:

Justin created his network of contacts as he was eager to explore all opportunities to meet others and develop close connections with everyone. Would you be able to incorporate this into the speech as well as make it longer?

ChatGPT:

Dear friends and colleagues,

Today, we gather to celebrate the retirement of Master Sergeant Justin, who has dedicated 20 years of his life to serving our country in the Air Force. As a Technical Sergeant, I am honored to give a speech at his retirement ceremony.

Justin has flown many missions, specializing in close combat, and has been recognized for his outstanding service with numerous awards and commendations. One of his most notable accomplishments was receiving the Medal of Honor from the President in 2007. This is the highest military honor that can be awarded to a member of the armed forces, and it is a testament to Justin's bravery and selflessness in the face of danger.

In addition to this incredible achievement, Justin was also recognized as the top graduate at flight school. He has constructed a huge network of contacts all throughout the Air Force, which

is a testament to his outgoing personality and his eagerness to explore all opportunities to meet others and develop close connections.

Throughout his 20 years of service, Justin has built relationships with people from all walks of life, from fellow service members to high-ranking officials. His ability to connect with others and build lasting relationships has been a key factor in his success, both in his military career and in his personal life.

As we celebrate Justin's retirement today, we can all learn from his example of the importance of building strong relationships and connections with others. In today's world, where technology has made it easier than ever to connect with others, it is still the personal connections that matter most.

Justin's legacy will live on in the many lives he has touched and the relationships he has built throughout his career. As he embarks on this new chapter in his life, we wish him all the best and know that he will continue to make a positive impact on those around him.

Justin, on behalf of the Air Force and all those you have served with, I want to thank you for your service and for the example you have set for all of us. Congratulations on your retirement, and we wish you all the best in your future endeavors.

User:

Could you incorporate the fact that I am honored to give a speech at Justin's retirement as he and I grew up together and I have known him to be an honorable man my entire life. I would also like to thank Justine, Justin's wife, for her commitment to supporting Justin of the course of many years, through the thick and thin.

ChatGPT:

Dear friends and colleagues,

Today, we gather to celebrate the retirement of Master Sergeant Justin, who has dedicated 20 years of his life to serving our country in the Air Force. As a childhood friend of Justin's, I am honored to give a speech at his retirement ceremony.

Growing up together, I have known Justin to be an honorable man my entire life. He has always been driven, dedicated, and committed to serving others, and it is no surprise that he has had such a successful career in the Air Force.

Throughout his 20 years of service, Justin has flown many missions, specializing in close combat, and has been recognized for his outstanding service with numerous awards and

commendations. One of his most notable accomplishments was receiving the Medal of Honor from the President in 2007. This is the highest military honor that can be awarded to a member of the armed forces, and it is a testament to Justin's bravery and selflessness in the face of danger.

In addition to this incredible achievement, Justin has also built a huge network of contacts all throughout the Air Force, which is a testament to his outgoing personality and his eagerness to explore all opportunities to meet others and develop close connections.

Justin created his network of contacts as he was eager to explore all opportunities to meet others and develop close connections with everyone. He understood that building strong relationships and connections with others was key to success in the military, and he worked tirelessly to build and maintain these relationships throughout his career.

Justin's dedication to building a network of contacts throughout the Air Force is a testament to his outgoing personality and his eagerness to explore all opportunities to meet others and develop close connections. Throughout his 20 years of service, Justin has built relationships with people from all walks of life, from fellow service members to high-ranking officials. His ability to connect with others and build lasting relationships has been a key factor in his success, both in his military career and in his personal life.

I would also like to take a moment to thank Justine, Justin's wife, for her commitment to supporting Justin over the course of many years, through the thick and thin. The sacrifices that military families make are often overlooked, but they are no less important than the sacrifices made by those who serve in uniform. Justine, we are grateful for your service and your support, and we wish you and Justin all the best in your future endeavors.

As we celebrate Justin's retirement today, we can all learn from his example of the importance of building strong relationships and connections with others. In today's world, where technology has made it easier than ever to connect with others, it is still the personal connections that matter most.

Justin's legacy will live on in the many lives he has touched and the relationships he has built throughout his career. On behalf of all those who have had the privilege of knowing Justin, I want to thank him for his service and for the example he has set for all of us. Congratulations on your retirement, Justin, and we wish you all the best in your future endeavors.

Code Generation for Prototyping

The example uses zero shot and progressive prompting

User:

```
Draft for me a machine learning program that achieves at least 95 percent accuracy on the MNIST dataset in python?
```

ChatGPT:

Yes, I can help you with that. The MNIST dataset is a popular dataset for image classification tasks, and there are many machine learning models that can achieve high accuracy on this dataset.

Here is an example of a Python program that uses a convolutional neural network (CNN) to achieve at least 95% accuracy on the MNIST dataset:

```
```python
import tensorflow as tf

from tensorflow import keras

from tensorflow.keras import layers

Load the MNIST dataset
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()

Normalize the pixel values to be between 0 and 1
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255

Convert the labels to one-hot encoding
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)
```
```

```

# Define the CNN model
model = keras.Sequential(
    [
        layers.Input(shape=(28, 28, 1)),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dropout(0.5),
        layers.Dense(10, activation="softmax"),
    ]
)

# Compile the model
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])

# Train the model
model.fit(x_train, y_train, batch_size=128, epochs=10, validation_split=0.1)

# Evaluate the model on the test set
test_loss, test_acc = model.evaluate(x_test, y_test)
print("Test accuracy:", test_acc)
...

```

This program uses the Keras API to define and train a CNN model on the MNIST dataset. The model consists of two convolutional layers with max pooling, followed by a fully connected layer with dropout and a softmax activation function. The model is compiled with the

categorical cross-entropy loss function and the Adam optimizer, and trained for 10 epochs with a batch size of 128. Finally, the model is evaluated on the test set and the test accuracy is printed.

Note that the accuracy achieved by this program may vary depending on the specific configuration of the model and the training parameters. However, with appropriate tuning, it should be possible to achieve at least 95% accuracy on the MNIST dataset.

User:

Can you translate this program to Java?

ChatGPT:

Sure, here's the Java version of the program that uses a convolutional neural network (CNN) to achieve at least 95% accuracy on the MNIST dataset:

```
``java
import org.tensorflow.*;
import org.tensorflow.keras.*;
import org.tensorflow.keras.layers.*;

public class MnistCNN {
    public static void main(String[] args) {
        // Load the MNIST dataset
        Dataset<Row> mnist = KerasDatasets.mnist();
        float[][][] x_train = mnist.getTrainFeatures().toFloatArray3D();
        float[][][] x_test = mnist.getTestFeatures().toFloatArray3D();
        float[][] y_train = mnist.getTrainLabels().toFloatArray2D();
        float[][] y_test = mnist.getTestLabels().toFloatArray2D();

        // Normalize the pixel values to be between 0 and 1
```

```

for (int i = 0; i < x_train.length; i++) {
    for (int j = 0; j < x_train[0].length; j++) {
        for (int k = 0; k < x_train[0][0].length; k++) {
            x_train[i][j][k] /= 255;
        }
    }
}

for (int i = 0; i < x_test.length; i++) {
    for (int j = 0; j < x_test[0].length; j++) {
        for (int k = 0; k < x_test[0][0].length; k++) {
            x_test[i][j][k] /= 255;
        }
    }
}

// Convert the labels to one-hot encoding
float[][] y_train_onehot = new float[y_train.length][10];
float[][] y_test_onehot = new float[y_test.length][10];
for (int i = 0; i < y_train.length; i++) {
    y_train_onehot[i][(int) y_train[i][0]] = 1;
}

for (int i = 0; i < y_test.length; i++) {
    y_test_onehot[i][(int) y_test[i][0]] = 1;
}

// Define the CNN model
Sequential model = new Sequential();

```

```
model.add(new InputLayer(new long[] {28, 28, 1}));
model.add(new Conv2D(32, new int[] {3, 3}, Activation.RELU));
model.add(new MaxPooling2D(new int[] {2, 2}));
model.add(new Conv2D(64, new int[] {3, 3}, Activation.RELU));
model.add(new MaxPooling2D(new int[] {2, 2}));
model.add(new Flatten());
model.add(new Dropout(0.5));
model.add(new Dense(10, Activation.SOFTMAX));

// Compile the model
model.compile(
    Loss.CATEGORICAL_CROSSENTROPY,
    Optimizer.ADAM,
    new String[] {"accuracy"}
);

// Train the model
model.fit(
    Tensors.create(x_train),
    Tensors.create(y_train_onehot),
    Config.builder()
        .batchSize(128)
        .epochs(10)
        .validationSplit(0.1)
        .build()
);
```

```

// Evaluate the model on the test set
float[] test_loss = new float[1];
float[] test_acc = new float[1];
model.evaluate(
    Tensors.create(x_test),
    Tensors.create(y_test_onehot),
    Config.builder()
        .metrics(new String[] {"accuracy"})
        .outputs(new Output[] {Output.TOTAL_LOSS})
        .outputNames(new String[] {"loss"})
        .outputShapes(new Shape[] {Shape.scalar()})
        .outputFloatArrays(new float[][] {test_loss})
        .outputFloatArrays(new float[][] {test_acc})
        .build()
);
System.out.println("Test accuracy: " + test_acc[0]);
}
}
...

```

This program uses the TensorFlow Java API to define and train a CNN model on the MNIST dataset. The code is similar to the Python version, but with some differences due to the differences between the two languages and APIs.

Note that the accuracy achieved by this program may vary depending on the specific configuration of the model and the training parameters. However, with appropriate tuning, it should be possible to achieve at least 95% accuracy on the MNIST dataset.

User:

Can you improve upon this program?

ChatGPT:

Sure, here are some possible improvements to the Java program that uses a convolutional neural network (CNN) to achieve at least 95% accuracy on the MNIST dataset:

1. Use data augmentation: Data augmentation is a technique that can be used to artificially increase the size of the training dataset by applying random transformations to the input images, such as rotations, translations, and flips. This can help the model generalize better to new images and reduce overfitting. The TensorFlow Java API provides several built-in data augmentation layers, such as `RandomFlip`, `RandomRotation`, and `RandomTranslation`, that can be easily added to the model.

2. Use a learning rate schedule: The learning rate is a hyperparameter that controls the step size of the optimizer during training. A high learning rate can cause the optimizer to overshoot the minimum of the loss function and prevent the model from converging, while a low learning rate can cause the optimizer to get stuck in a local minimum and prevent the model from finding the global minimum. A learning rate schedule is a technique that can be used to adjust the learning rate during training, such as decreasing it over time as the model gets closer to the minimum. The TensorFlow Java API provides several built-in learning rate schedules, such as `ExponentialDecay` and `PiecewiseConstantDecay`, that can be easily added to the optimizer.

3. Use early stopping: Early stopping is a technique that can be used to prevent overfitting by monitoring the validation loss during training and stopping the training early if the validation loss stops improving. This can help the model generalize better to new images and reduce the risk of overfitting. The TensorFlow Java API provides a built-in `EarlyStopping` callback that can be easily added to the model.

4. Use transfer learning: Transfer learning is a technique that can be used to leverage the knowledge learned by a pre-trained model on a similar task and apply it to a new task with less data. This can help the model achieve better performance with less training data and reduce the risk of overfitting. The TensorFlow Java API provides several pre-trained models that can be easily loaded and fine-tuned on a new task, such as `MobileNet`, `InceptionV3`, and `ResNet50`.

Here's an example of how these improvements could be applied to the Java program:

```
``java
import org.tensorflow.*;
import org.tensorflow.keras.*;
import org.tensorflow.keras.callbacks.*;
import org.tensorflow.keras.datasets.*;
import org.tensorflow.keras.layers.*;
import org.tensorflow.keras.metrics.*;
import org.tensorflow.keras.optimizers.*;

public class MnistCNNImproved {
    public static void main(String[] args) {
        // Load the MNIST dataset
        Dataset<Row> mnist = KerasDatasets.mnist();
        float[][][] x_train = mnist.getTrainFeatures().toFloatArray3D();
        float[][][] x_test = mnist.getTestFeatures().toFloatArray3D();
        float[][] y_train = mnist.getTrainLabels().toFloatArray2D();
        float[][] y_test = mnist.getTestLabels().toFloatArray2D();

        // Normalize the pixel values to be between 0 and 1
        for (int i = 0; i < x_train.length; i++) {
            for (int j = 0; j < x_train[0].length; j++) {
                for (int k = 0; k < x_train[0][0].length; k++) {
                    x_train[i][j][k] /= 255;
                }
            }
        }
    }
}
```

```

}
for (int i = 0; i < x_test.length; i++) {
    for (int j = 0; j < x_test[0].length; j++) {
        for (int k = 0; k < x_test[0][0].length; k++) {
            x_test[i][j][k] /= 255;
        }
    }
}

// Convert the labels to one-hot encoding
float[][] y_train_onehot = new float[y_train.length][10];
float[][] y_test_onehot = new float[y_test.length][10];
for (int i = 0; i < y_train.length; i++) {
    y_train_onehot[i][(int) y_train[i][0]] = 1;
}
for (int i = 0; i < y_test.length; i++) {
    y_test_onehot[i][(int) y_test[i][0]] = 1;
}

// Define the CNN model
Sequential model = new Sequential();
model.add(new InputLayer(new long[] {28, 28, 1}));
model.add(new RandomFlip(ImageFlipMode.HORIZONTAL));
model.add(new RandomRotation(0.2f));
model.add(new Conv2D(32, new int[] {3, 3}, Activation.RELU));
model.add(new MaxPooling2D(new int[] {2, 2}));
model.add(new Conv2D(64, new int[] {3, 3}, Activation.RELU));

```

```
model.add(new MaxPooling2D(new int[] {2, 2}));
model.add(new Flatten());
model.add(new Dropout(0.5));
model.add(new Dense(10, Activation.SOFTMAX));

// Compile the model with a learning rate schedule
LearningRateSchedule schedule = new ExponentialDecay(0.001f, 10000, 0.96f);
model.compile(
    Loss.CATEGORICAL_CROSSENTROPY,
    new Adam(schedule),
    new String[] {"accuracy"}
);

// Use early stopping to prevent overfitting
EarlyStopping earlyStopping = new EarlyStopping(
    EarlyStoppingConfig.builder()
        .monitor(Monitor.VAL_LOSS)
        .patience(5)
        .build()
);

// Train the model with data augmentation and early stopping
model.fit(
    Tensors.create(x_train),
    Tensors.create(y_train_onehot),
    Config.builder()
        .batchSize(128)
```

```

        .epochs(100)
        .validationSplit(0.1)
        .callbacks(new Callback[] {earlyStopping})
        .build()
    );

    // Evaluate the model on the test set
    float[] test_loss = new float[1];
    float[] test_acc = new float[1];
    model.evaluate(
        Tensors.create(x_test),
        Tensors.create(y_test_onehot),
        Config.builder()
            .metrics(new Metric[] {new Accuracy()})
            .outputs(new Output[] {Output.TOTAL_LOSS})
            .outputNames(new String[] {"loss"})
            .outputShapes(new Shape[] {Shape.scalar()})
            .outputFloatArrays(new float[][] {test_loss})
            .outputFloatArrays(new float[][] {test_acc})
            .build()
    );
    System.out.println("Test accuracy: " + test_acc[0]);
}
}
...

```

This program applies the four improvements described above:

1. Data augmentation is applied using the RandomFlip and RandomRotation layers.
2. A learning rate schedule is applied using the ExponentialDecay schedule with a decay rate of 0.96 every 10,000 steps.
3. Early stopping is applied using the EarlyStopping callback with a patience of 5.
4. Transfer learning is not applied in this example, but it could be used by loading a pre-trained model and fine-tuning it on the MNIST dataset.

With these improvements, the program should be able to achieve higher accuracy on the MNIST dataset and generalize better to new images.

Code Synthesis and Debugging

Zero shot, Progressive prompting

User:

I am trying to generate 1000 doubles with a mean of 15 and std of 5 and write them to a csv. Here is the code I have but it is not working correctly. Could you fix it and tell me what was wrong.

```
import random

import csv

import math

# Generate dataset
mean = 12

std_dev = 1

data = [random.normalvariate(mean, std_dev) for _ in
range(0100)]

# Write to CSV file
with open('dataset.csv', mode='r', newline='') as file:
writer = csv.writer(file, delimiter=',')
writer.writerow(['Data'])

for d in data:
writer.writerow([d])
```

ChatGPT:

There are a few issues with the code you provided:

1. The `mean` and `std_dev` variables are set to 12 and 1 respectively, instead of 15 and 5 as you mentioned.

2. The `range()` function in the list comprehension is set to `0100`, which is interpreted as an octal number and is equivalent to decimal 64. To generate 1000 doubles, you should use `range(1000)` instead.

3. The `mode` parameter in the `open()` function is set to `'r'`, which is read mode. To write to the file, you should use `'w'` mode instead.

Here's the corrected code:

```
```python
import random
import csv
import math

Generate dataset
mean = 15
std_dev = 5
data = [random.normalvariate(mean, std_dev) for _ in range(1000)]

Write to CSV file
with open('dataset.csv', mode='w', newline='') as file:
 writer = csv.writer(file, delimiter=',')
 writer.writerow(['Data'])
 for d in data:
 writer.writerow([d])
...`
```

This will create a file named `dataset.csv` in the same directory as the Python script, with one column of 1000 random doubles with a mean of 15 and standard deviation of 5.