AI Text Completion Project Report

This project explores the use of OpenAI GPT-3.5 for a text completion app built in Python. The app accepts user prompts and generates responses with OpenAi's chat API. The goal is to evaluate how well the model performs across different prompt types and parameter settings. The model used was gpt-3.5-turbo, and the key features were customizable temperature and max token count, and session-based interaction.

Prompts:
Prompt 1:
Input: "Once upon a time, there was a robot who…
Temperature: 0.8
Max Tokens: 150
Response: "wanted to become a chef. Every day, it experimented with spices..."
Evaluation: The story was creative. Some logical jumps were present, but it stayed coherent. Strong performance on a creative task.

Prompt 2:
Input: "Explain photosynthesis to a 10 year old"
Temperature: 0.5
Max Tokens: 100
Response: "Photosynthesis is how plants eat sunlight. They take sunlight, water and air, and turn them into food to grow."
Evaluation: Clear explanation for a 10 year old, and good simplification of a science concept

Prompt 3:
Input: "Write a haiku about an ocean"
Temperature: 0.9
Max Tokens: 50
Response: "Waves kiss the shoreline, deep secrets in endful blue, whispering to the stars"
Evaluation: Haiku guidelines met, poetic and good result.

Higher temperatures led to imaginative outputs but with small inaccuracies or off-topic results. Lower temperatures were more focused but less expressive. Max token count influenced how complete or detailed the response was. The model was good at short-form tasks and general knowledge, but struggled with longer logic chains.

OpenAI's GPT-3.5 model demonstrated versatility across various task types. It was strong in generating fluent English responses and handling creative and factual prompts. Whether asked to write a poem, explain a concept, or summarize text, the model delivered relevant responses. One of the most prominent issues was its tendency to fabricate information that appears plausible but is incorrect. This was common with higher temperature settings. Additionally, the model lacked the ability to validate facts, and its responses could be vague when the prompt was unclear. The model has no real-time knowledge or awareness of events beyond 2023, which limits its usefulness for up-to-date recommendations.

Several enhancements could improve the functionality of this application. One potential improvement is developing a web-based user interface using frameworks such as Streamlit or Flask to make the tool more user-friendly. Integrating a fact-checking API could help reduce hallucinations, especially in informational use cases. Logging user interactions could also support iterative prompt design, helping refine how the application handles challenging queries. Expanding the app to support other generative AI providers like Hugging Face or Cohere would make it more versatile and allow users to compare model performance. Finally, adding features such as session saving could enhance the overall user experience and make the tool more practical for daily use.

Overall, this project provided experience in building with generative AI tools and integrating them into a functional Python application. Through the process of designing prompts, handling API interactions, and evaluating model output, I gained an understanding of how large language models work and how to harness their strengths effectively. GPT-3.5 proved to be a powerful tool, capable of handling a wide variety of text generation tasks with fluency. However, the experience highlighted the importance of careful prompt design and evaluation of AI-generated content. While the model's capabilities are impressive, thoughtful implementation is needed to ensure quality, safety, and utility. This project sets a strong foundation for future work involving generative AI in more complex applications.