



# GPT Summarizer

Alyssa Zhu, Alex Tang, Nathan Papa



---

# Motivation and Goals

- **Project Objective:** Address the challenge of students struggling to read and extract key information from class material
  - **Solution:** Create text summarizer feature using OpenAI GPT API
    - Prompt Engineering:
      - Develop prompts for generating concise and effective summaries
      - Test out different models and parameters
    - React + NodeJS App:
      - Connect OpenAI GPT NodeJS API to frontend
-



# Prompt Engineering

# ChatGPT Prompt Engineering

- ChatGPT API provides several helpful fields for prompt engineering
  - n - How many responses ChatGPT generates per prompt
    - Limit output to a finite amount of bulletpoints/sentences
  - temperature - randomness in the generation of output
    - Higher temperature limits the chances of repeated thoughts in our summary
  - presence\_penalty - penalizes tokens that have appeared before
    - Motivates our chatbot to discuss things it hasn't before
  - messages - Allows us to tell ChatGPT what we specifically want from it
    - "Summarize this input in this way."

---

# Core Ideas

- How long should our summary be?
  - Longer summaries defeat the purpose of summarizing the text but shorter summaries might miss crucial details
- We decided to generate shorter summaries
  - While also guaranteeing a much quicker read than the input text, shorter summaries also are less repetitive
- How to scale length?
  - Limit each output to a fixed word length and then scale the number of outputs to the length of the input

---

# Algorithm

1. Calculate input length and determine number of sentences or bullet points to generate based off of that.
2. Use messages field to inform our chatbot of the output's format and word length.
3. Minimize repetition with presence\_penalty and temperature.
4. Concatenate the output and return it to the output textbox


---

# React & NodeJS App

---

# Summarizer Page

- Text is displayed within textareas
  - User input
  - Summary (Read-only)
- User can select summary type
  - Paragraph
  - Bullet Points



## Text Summarizer

Paste Text Here:

Summary:

It's Just Adding One Word at a Time  
That ChatGPT can automatically generate something that reads even superficially like human-written text is remarkable, and unexpected. But how does it do it? And why does it work? My purpose here is to give a rough outline of what's going on inside ChatGPT—and then to explore why it is that it can do so well in producing what we might consider to be meaningful text. I should say at the outset that I'm going to focus on the big picture of what's going on—and while I'll mention some engineering details, I won't get deeply into them. (And the essence of what I'll say applies just as well to other current "large language models" [LLMs] as to ChatGPT.)

The first thing to explain is that what ChatGPT is always fundamentally trying to do is to produce a "reasonable continuation" of whatever text it's got so far, where by "reasonable" we mean "what one might expect someone to write after seeing what people have written on billions of webpages, etc."

So let's say we've got the text "The best thing about AI is its ability to". Imagine scanning billions of pages of human-written text (say on the web and in digitized books) and finding all instances of this text—then seeing what word comes next what fraction of the time. ChatGPT effectively does something like this, except that (as I'll explain) it doesn't look at literal text; it looks for things that in a certain sense "match in meaning". But the end result is that it produces a ranked list of words that might follow, together with "probabilities":

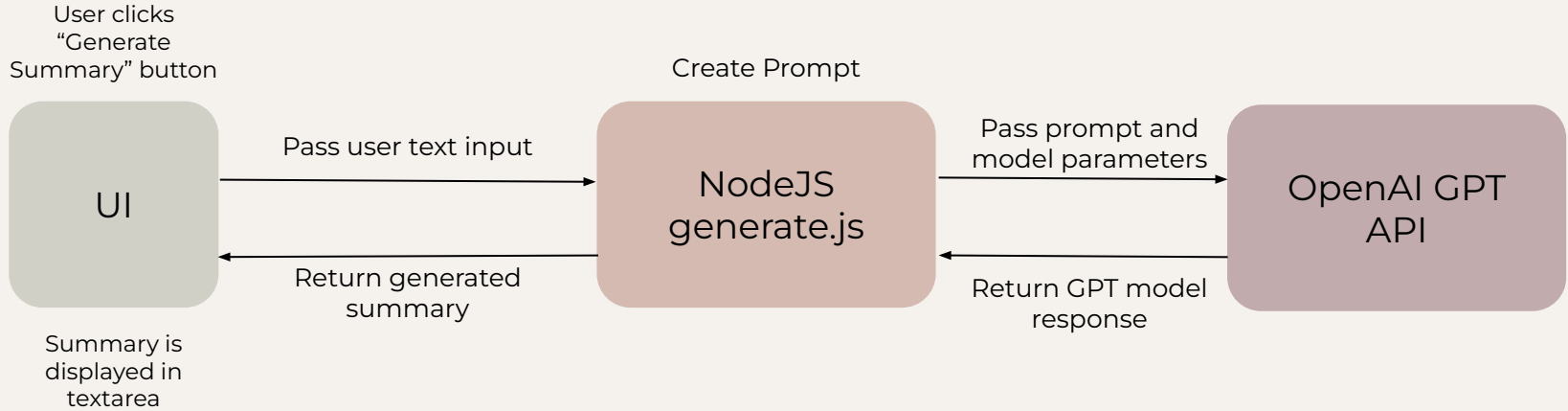
- ChatGPT generates human-like text by attempting to produce a "reasonable continuation" of the given text.  
- It scans billions of human-written text to determine the most probable next word based on semantic matching and produces a ranked list of words with probabilities.

Summary Type:  Paragraph  Bulleted

Generate Summary



# API Call Structure



The image features two horizontal lines, one at the top and one at the bottom. Each line has a smooth, curved end on the left and right sides, creating a frame-like effect. The word "Demo" is centered between these lines.

**Demo**

The image features a minimalist design with two horizontal lines, one at the top and one at the bottom. Each line has a smooth, curved end that extends slightly beyond the frame. The word "Testing" is centered between these lines in a large, bold, black serif font.

# Testing

# Two Examples

- From a content standpoint, our app performs quite well
  - The bottom test gives key words but not coherent sentences
  - The top provides a response similar to ours
- Most examples online gave pretty good ideas on other features to add

The screenshot shows a web-based summarization tool. At the top, there are tabs for 'Paragraph', 'Key Sentences', and 'Summary Length: Short'. A slider is positioned between 'Short' and 'Long'. The main content area is split into two columns. The left column contains the original text, and the right column contains the summarized text. Below the text, there is a 'Select keywords' section with a search icon and a dropdown arrow. Below that, there are several keyword tags: 'hardware concurrency', 'Sun Solaris', 'threads', 'process', and 'level'. At the bottom left, it says '86 words'. At the bottom center, there is a green 'Summarize' button. At the bottom right, it says '2 sentences • 35 words' and a 'Paraphrase Summary' button with icons for copy, download, and print.

The screenshot shows a desktop application interface for summarization. At the top, there is a 'Summarizer' tab and an 'AI Summarizer' section with a slider set to '50%'. The main area is split into two columns. The left column contains the original text, and the right column contains the summarized text. At the bottom left, there is an upload icon. At the bottom center, there is a language dropdown set to 'English'. At the bottom right, there are buttons for 'Check Plagiarism', 'Paraphrase', and 'Show Bullets', along with icons for copy, download, and print.



# **Future Improvements**

---

# Future Improvements

- Summary generation:
    - More variety across bullet points/sentences in generated summaries
    - Ability to adjust response length
    - Add support for other input formats (image and text files)
    - Upgrade to GPT-4
  - Question generation: add ons to the original response to learn more information
  - Add feature to React Native App
-