Co-writing with AI language models

Exercises

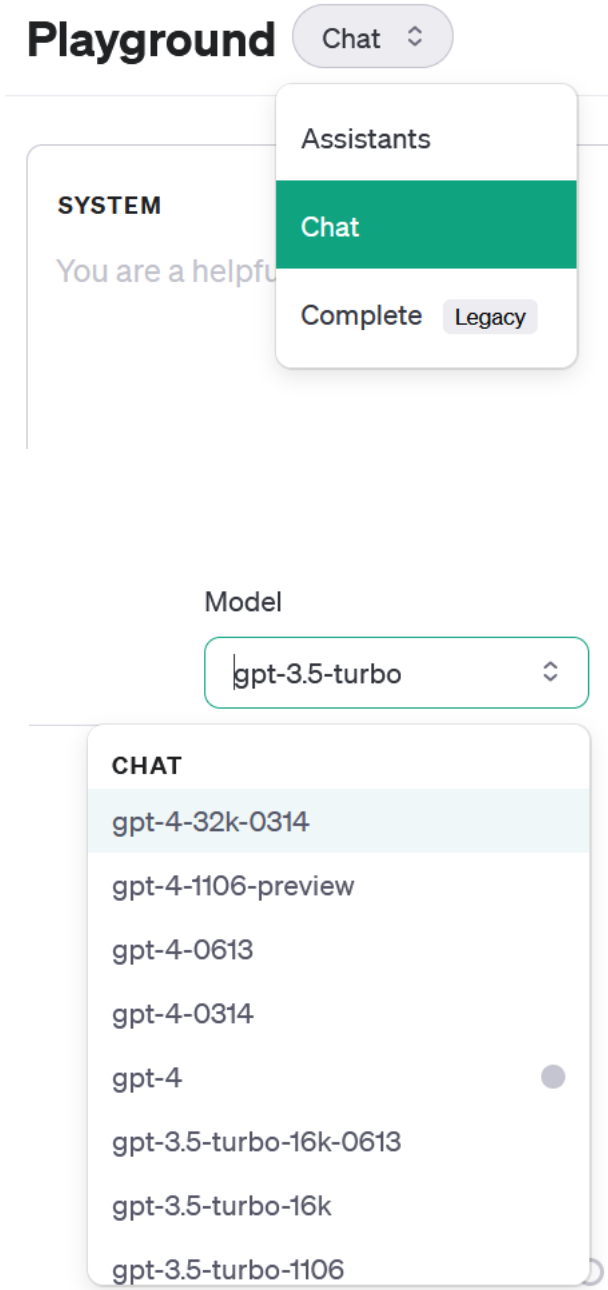
Prof. Perttu Hämäläinen, Jan 2024

Tools

- For everyone:
  - OpenAI's GPT series models (Bing chat, Microsoft CoPilot mobile app, ChatGPT, OpenAI Playground, https://platform.openai.com)
  - Anthropic Claude 2, https://www.anthropic.com/index/claude-2
  - Write with Laika (GPT-2 models finetuned with specific public domain literary works, e.g., Kafka) https://www.writewithlaika.com/
- For programmers
  - OpenAI API
  - Amazon Bedrock
  - Huggingface Transformers library

General guidance

- Try and compare different models to get the feel of their strengths and weaknesses
  - GPT-4 is very knowledgeable but has a specific default “voice” and low variablility in responses, davinci-002 requires more examples to work well but has higher variability & creativity
- Remember the 2 principles: be specific and give concrete examples
- Try and practice well established prompting techniques: https://platform.openai.com/docs/guides/prompt-engineering



Exercise: Generate game ideas (Level 1)

- Learning goal: See the effect of examples and practice finding & writing good examples
- Compare these ways of prompting with ChatGPT:
  1. “Please give me an innovative game idea”
  2. “Please add more innovative game ideas to the list below. All ideas must have a clear design hook that makes players want to wishlist and try the game.” Then add one or more examples before sending the prompt to the AI.
  3. Same as above, but after the model has produced many examples, ask it to recommend the one that is the most novel, exciting, and likely to compel players to wishlist and buy the game.

Try to write the examples by describing your favorite award-winning or otherwise successful games. Or copy-paste from Steam’s most wishlisted game descriptions.

Variation: generate book opening sentences or quotes

- If games are not your thing, you can use the same approach with examples from known books. Google for “best opening sentences” or check Amazon’s most highlighted Kindle text segments

Exercise: Generate game ideas (Level 2)

- Learning goal: Practice constructing prompts in code and automating tedious manual prompting. Choose one:
  1. Use OpenAI API or equivalent (in Colab or locally on your computer)
  2. Use an open source LLM locally: Llama 2 7B, Mistral, or Microsoft Phi. Google “mistral colab” or “phi llm colab” to get started
- Scrape reference game descriptions, e.g., from Steam. Google “python web scraping” or ask ChatGPT how
- In your code, select a random sample of the descriptions and construct the prompt, starting with “Please add more innovative game ideas to the list below. All ideas must have a clear design hook that makes players want to wishlist and try the game.”
- Send the prompt to the LLM
- Construct another prompt asking the model to select the best idea, starting e.g., “Which of the game ideas below is the most innovative and likely to compel players to wishlist and buy?”

Exercise: Visualize text (Level 2)

- Learning goal: Practice using embeddings to explore and visualize text data.
- Starting point: 100 game ideas or opening sentences, either generated in the previous exercise or scraped from the Internet
- Compute embedding vectors for the texts using either OpenAI API or Sentence Transformers (google “sentence transformers embeddings colab”
- Reduce the dimensionality of the embeddings to 2 or 3 for visualization using PCA, UMAP, or MDS (google e.g. “python dimensionality reduction UMAP”)
- Create an interactive scatterplot visualization using Plotly’s Scatter function (google “plotly scatter colab”). Example: https://github.com/PerttuHamalainen/LLMCode/blob/master/test_results/bopp_test_visualization.gif

Exercise: Retrieval-augmented generation (Build your own ChatPDF, levels 2+)

- Retrieval-augmented generation means that the user can ask questions about some corpus of text, e.g., a pdf of a book
  1. Preprocess the corpus by computing embeddings of each paragraph
  2. Preprocess the question by computing its embedding
  3. Construct a prompt that includes paragraphs with the lowest cosine distance between the question and the paragraph embeddings
  4. Make a query to a LLM using the prompt

  There are multiple details to get right here, but a basic version is doable in a short time. This can also become your final project.

  Challenge: Also plug in a speech-to-text model such as OpenAI Whisper. Using it, create your data by transcribing a podcast such as https://www.idlethumbs.net/designernotes/