Going beyond ChatGPT: an introduction to prompt engineering and LLMs

PyData Seattle – 2023

Ties de Kok

UNIVERSITY of WASHINGTON

FOSTER SCHOOL OF BUSINESS

A bit about me

Ties de Kok

Assistant Professor in Accounting
University of Washington

About me | Research | Code & Data | Talks & Classes | Blog Posts | CV

Hi there! 👋 I am an Assistant Professor at the University of Washington, Foster School of Business. I specialize in combining computer science with accounting research.

My research interests include:

- Machine Learning
- Natural Language Processing
- Social Media
- ESG Disclosures
- XBRL
- And much more! 🚀

Ties de Kok
TiesdeKok

Follow

Assistant Professor in Accounting at the University of Washington. I specialize in combining computer science with empirical Accounting research.

Overview | Repositories 30 | Projects | Packages | Stars 257

Pinned

LearnPythonforResearch Public
This repository provides everything you need to get started with Python for (social science) research.
Jupyter Notebook 232 81

Python_NLP_Tutorial Public
This repository provides everything to get started with Python for Mining / Natural Language Processing (NLP)
Jupyter Notebook 107 60

ipystata Public
Enables the use of Stata together with Python via Jupyter (IPython) notebooks.
Jupyter Notebook 183 64

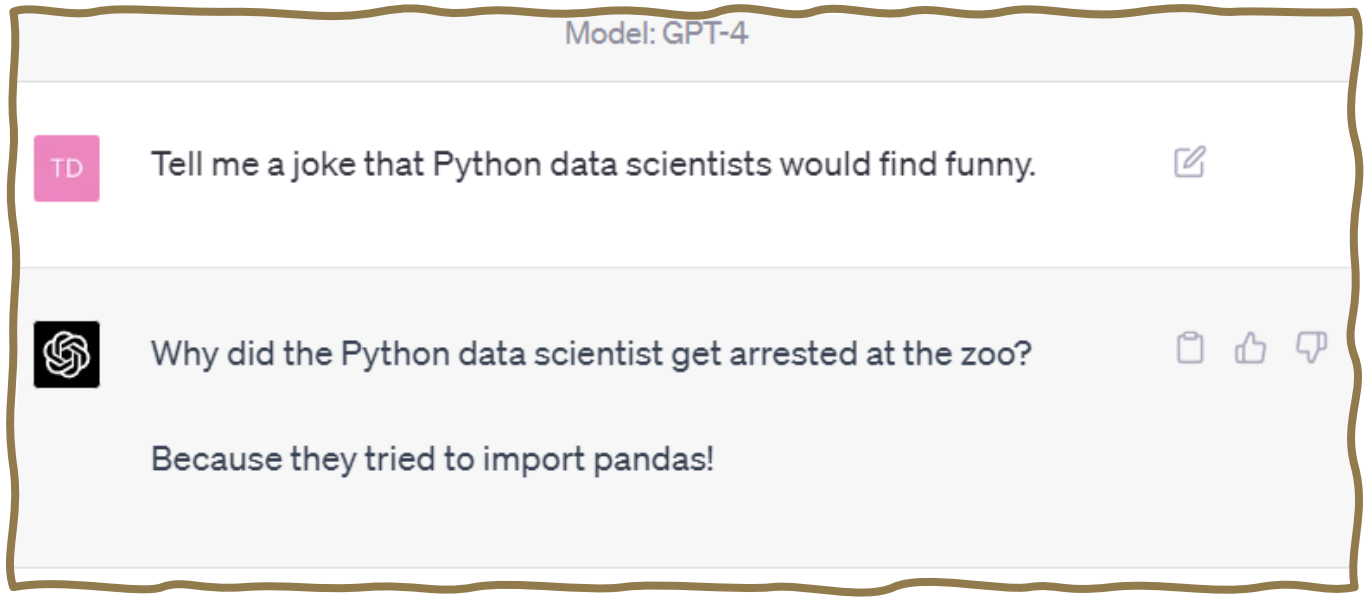
fast_xbrl_parser Public
An XBRL parser built in Rust that provides a fast, easy, and lightw to convert XBRL XML files into JSON or CSV.
Rust 13 7

gpt-prompt-example Public
This is a companion repository to my blog post contain Python code to show how to do zero shot, few shot, and fine-tuning using the OpenAI API.
Jupyter Notebook 6 2

chat-gpt-jupyter-extension Public
A browser extension to provide various AI helper functions in Jup Notebooks, powered by ChatGPT.
TypeScript 177 20

Tutorial overview

Generative LLMs (such as ChatGPT) are really cool:



#1 Uses of GLLMs for data scientists?
#2 How do we use GLLMs programmatically?

My background

My day-to day:

Research:

➔ Textual analysis and ML to study capital markets

Teaching:

➔ Python & accounting analytics

tds Published in Towards Data Science

Ties de Kok

Apr 16, 2022 · 11 min read · Listen

Beyond chat-bots: the power of prompt-based GPT models for downstream NLP tasks

https://medium.com/towards-data-science/beyond-chat-bots-the-power-of-prompt-based-gpt-models-for-downstream-nlp-tasks-21eff204d599

Generative LLMs and Textual Analysis in Accounting: (Chat)GPT as Research Assistant?

April 2023

https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4429658

Tutorial structure

Part 1: introduction to generative LLMs (GLLMs)

Part 2: deep-dive example on automating ChatGPT using the OpenAI API

Part 3: what is possible? ➔ LangChain demonstration

All session materials:

➔ *https://github.com/TiesdeKok/pydata_2023*

A few disclaimers

#1 The GLLM space is fast developing.
When I submitted my proposal for this talk in Feb. 2023 all the following did not exist (publicly):

➔ GPT-4, ChatGPT API, LLaMa, Alpaca, and more

#2 My primary GLLM experience is in academia

➔ e.g., I never have to take things into production

#3 We will only interact with the OpenAI API

➔ Many other options, but we only have 90 minutes. ☺

Want to follow along?

All the session materials:

https://github.com/TiesdeKok/pydata_2023

Binder link:

To get started with the demonstration portion, click the button below!

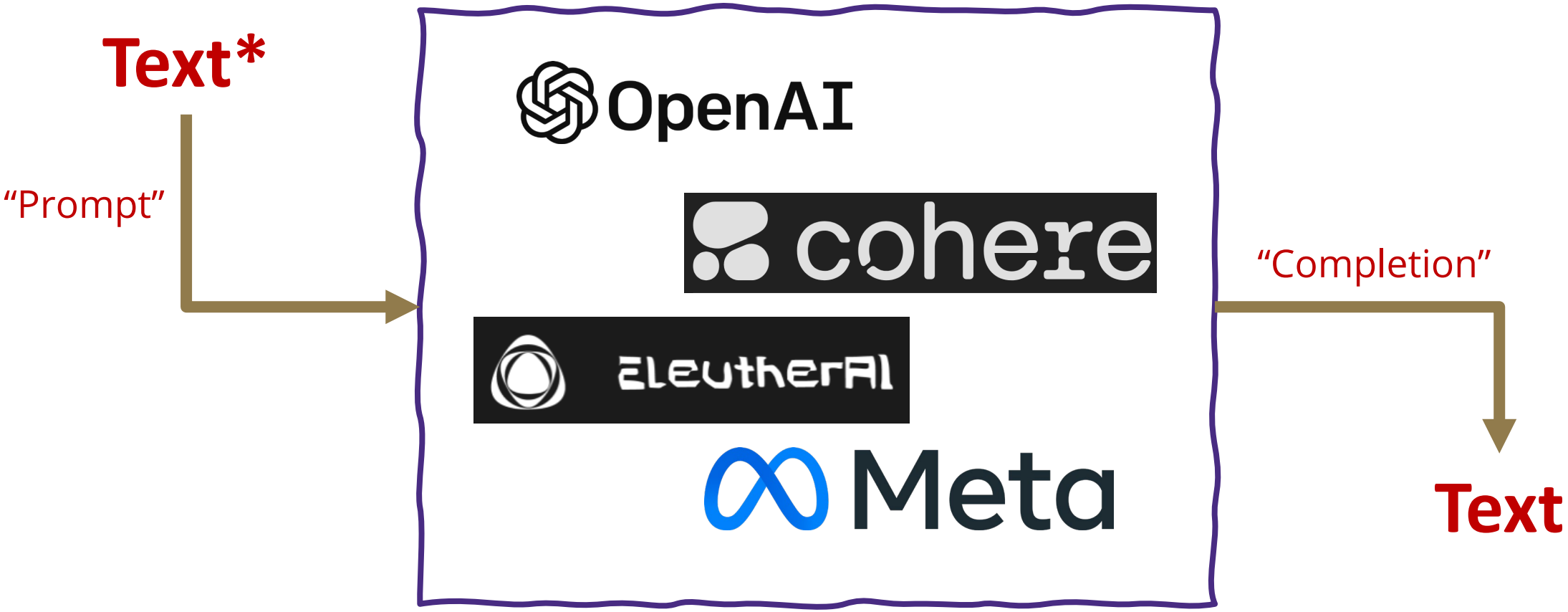
launch binder

To run the code, you will need an OpenAI API key → See the steps in the notebook on how to obtain one.

Part 1:
Introduction to GLLMs

What is a Generative Large Language Model?

GLLMs → generate text based on text*



What is a Generative Large Language Model?

Tell me a joke that Python data scientists would find funny

OpenAI

cohere

EleutherAI

Meta

Why did the Python data scientist get arrested?

Because they tried to import Pandas!

What uses does this have?

Why are GLLMs so useful?

Key benefit #1:

Task are communicated to GLLMs using just text.

Key benefit #2:

GLLMs can often handle tasks with little to no training.

Primary benefit: GLLMs can substitute for manual work or complex machine learning pipelines

Working example

- We are data scientists at a company with a mobile app.
- Customers complaints ➔ app is slow and unresponsive.
- Developers pushed a new update to fix this.
- **Our job:** *did the update work?*

1. *It's annoying when the app takes so long to process a payment.*
2. *The app could benefit from a better recommendation system for products.*
3. *I appreciate the recent update, but please add more filter options when searching for items.*
4. *The app often becomes unresponsive, especially when browsing through the product list.*

➔ How do we quantify this at scale?

Traditional solutions

Do it manually, or outsource it

Develop an NLP approach or machine learning

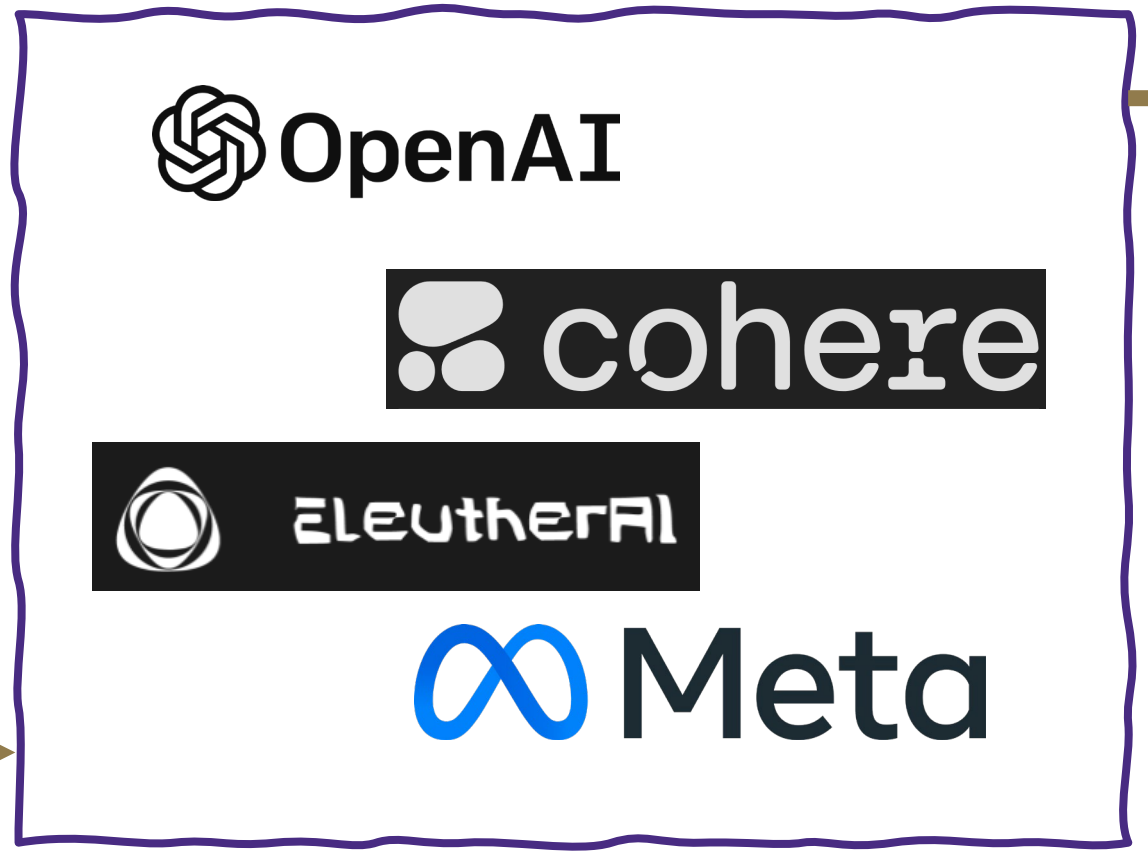
The issue: for many tasks this is going to be too slow, time-consuming, and expensive….

GLLM solution

Does the feedback mention performance issues?

{ feedback }

JSON =



{"answer" : 0}

Prompt "engineering"

Basic steps:

> **Figure out approach & model**
  - **Approaches:** zero shot, few-shot, fine-tuning
  - **Models:** OpenAI / Cohere / LlaMa / many more

> **Develop prompt (*"prompt engineering"*)**
  - Interacting with a GLLMs is a weird mix between talking to a human and interacting with a machine.

> **Evaluate output**

Prompt-engineering tips

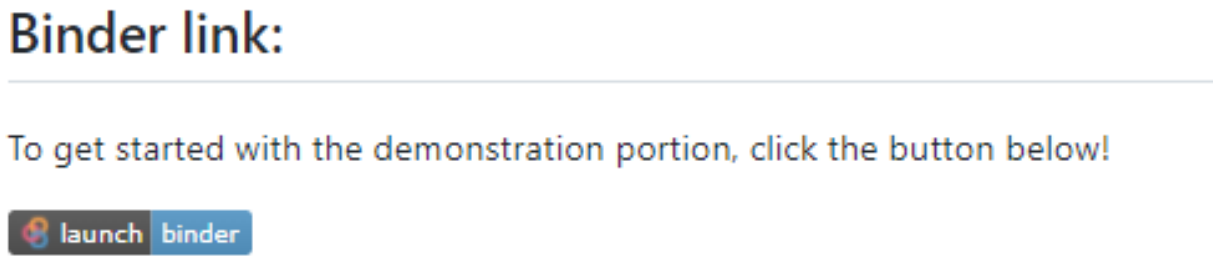
- **GLLMS generate text left-to-right → so its own generations become part of the prompt.**
  - *Asking the model to explain itself can make a big difference*
- ***Optimize the number of tokens***
  - *Number of tokens == speed and $$$*
- ***Be specific and to the point***
  - *GLLMs can have a hard time ignoring information*
- ***Small prompt differences can make a big difference***
- ***Prompt won't work? → Smaller model + fine-tuning***

See Appendix D in my paper for more details

Part 2:
Basic automations using ChatGPT API

Basic GLLM automation

> Let's automate the customer feedback problem!



https://github.com/TiesdeKok/pydata_2023

Part 3:

What is possible?
LangChain demo

Prompt-chaining

A single prompt can do powerful things

➔ **But what if we chain them together?**

Auto-GPT: An Autonomous GPT-4 Experiment

unit tests passing | AutoGPT 24931 members | Stars 112k | Follow @siggravitas

Auto-GPT is an experimental open-source application showcasing the capabilities of the GPT-4 language model. This program, driven by GPT-4, chains together LLM "thoughts", to autonomously achieve whatever goal you set. As one of the first examples of GPT-4 running fully autonomously, Auto-GPT pushes the boundaries of what is possible with AI.

https://github.com/Significant-Gravitas/Auto-GPT

To illustrate:

Let's say we want to calculate monthly sales using SQL

- **User -> GLLM:** what SQL table and column contains the sales transactions and amounts?
- **GLLM:** The GLLM checks the SQL metadata ➔ but can't find it because all the tables and columns are cryptic acronyms .
- **GLLM → GLLM**: Check the documentation for the table + column. ➔ Which yields a table + column.
- **GLLM → GLLM**: Check whether the table + column from the documentation exist in the actual SQL tables. ➔ Yes!
- **GLLM → User:** the table and column are ....

Easiest way to do this using Python?

🦜🔗 LangChain

hwchase17 / langchain Public

29.9k stars

323 watching

3k forks

https://github.com/hwchase17/langchain

🤔 What is this?

Large language models (LLMs) are emerging as a transformative technology, enabling developers to build applications that they previously could not. But using these LLMs in isolation is often not enough to create a truly powerful app - the real power comes when you can combine them with other sources of computation or knowledge.

This library is aimed at assisting in the development of those types of applications. Common examples of these types of applications include:

❓ Question Answering over specific documents

- Documentation
- End-to-end Example: Question Answering over Notion Database

💬 Chatbots

- Documentation
- End-to-end Example: Chat-LangChain

🤖 Agents

- Documentation
- End-to-end Example: GPT+WolframAlpha

LangChain demonstration

Binder link:

To get started with the demonstration portion, click the button below!

launch binder

https://github.com/TiesdeKok/pydata_2023

Concluding remarks

Things to be mindful of

- The performance of GLLMs can be deceiving
- Training data limitations and concerns:
  - Biases
  - Knowledge cut-off
  - Attribution concerns
- Data privacy and proprietary concerns
  - Try asking ChatGPT what it knows about you → any data you give it might surface in the same way if you are not careful!

Using ChatGPT for coding?

There are many projects that help to integrate GLLMs into Jupyter.

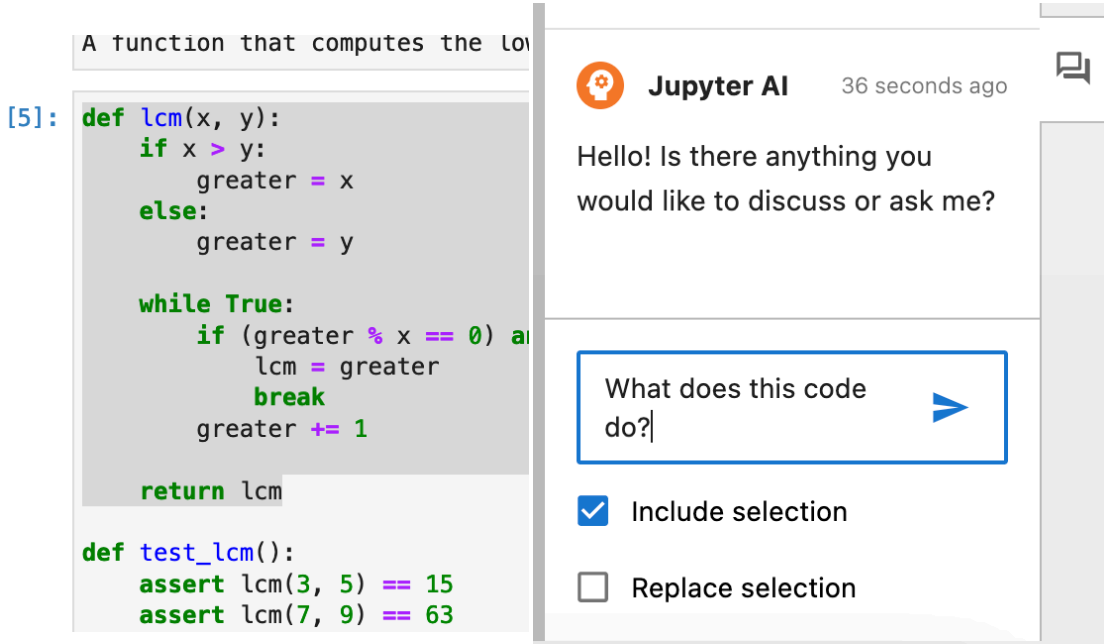
TiesdeKok / **chat-gpt-jupyter-extension** Public

A browser extension to provide various AI helper functions in Jupyter Notebooks, powered by ChatGPT.

https://github.com/TiesdeKok/chat-gpt-jupyter-extension

jupyterlab / **jupyter-ai** Public

A generative AI extension for JupyterLab



https://github.com/jupyterlab/jupyter-ai

Want to learn more?

Follow me on GitHub:

https://github.com/TiesdeKok

Check out my paper & GitHub repository:

https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4429658

https://github.com/TiesdeKok/gllm_companion

Questions?

Thank you!

Enjoy the rest of the conference ☺

UNIVERSITY *of* WASHINGTON

FOSTER SCHOOL OF BUSINESS