

A User's Guide to GPT and LLMs for Economic Research

Kevin Bryan

Based on the author's [Markus' Academy talk](#) - May 2023

1 Introduction

- Large language models (“LLMs”) are novel: a team at Google in 2017 introduced the fundamental insights in their paper “Attention is All You Need” ([Vaswani et al. 2017](#)). Astonishing results have come with GPT3 in 2021 and Dall-E/Stable Diffusion (image generators) in 2022.
- LLMs have been shown to increase productivity in sales and programming in the past ([Kalliamvakou 2022](#)). Academia should capitalize on this technology. This document tries to serve as an introductory practical guide for economists to do so. The main takeaways:
 1. Controlling the output of LLMs is difficult
 2. The “Raw” ChatGPT online is far from state of the art
 3. Hallucinations are mostly fixable (this guide includes some tips to avoid them)
 4. The technology's rate of improvements is fast
 5. You should get access to the API. Most use cases for economists require using the API and accompanying with code. This will give you much more control on the output, and it is cheap to do so
- This guide is primarily meant for research, but if you are interested, you can also review my guide on how to make a “[Virtual TA](#)” for your classes.

2 Understanding how LLMs work to control their output

- In a nutshell, LLMs take a huge corpus of written language to predict the next word in a sentence. The model tries to adjust the posterior distribution of words that will follow, based on language that already exists, to produce a response. This is why LLMs are called “stochastic parrots.”
- For example, suppose we tried to predict the next word after “The sun...”. Without further context, many options are possible (“is”, “shines”, “rose”...). How could we prevent it from yielding the most common output? Here are four ways to do so. Keep in mind our goal is always to shift that posterior distribution.
 1. **Zero-shot prompt engineering:** we can change our prompt to elicit a particular response. Suppose we ask the LLM to predict the next word after “The sun”, while also telling it: “Let me hear a story. Tell me about the weather yesterday.” This will be sufficient so shift the language to the past tense (“The sun **was**”). Cheap and easy, but often imprecise for research purposes.
 2. **Zero-shot context definition:** we can tell it that the context of our conversation are stories written by Hemingway. This might yield “The sun **also rises**”, which would otherwise be a very unlikely output.

3. **Few-shot context definition:** we can give it even more context, giving it examples of how we would like it to respond. For example, after showing it: “Examples: User: The moon. System: is 0.24 million miles from Earth. User: Mars. System: is 140 million miles from Earth. User: Pluto. System: is 4.67 billion miles from Earth”, we can input the “The sun” to obtain “is 93 million miles from Earth”. This kind of prompting is the main way industry tries to control the output of LLMs.
 4. **Fine-tuning:** We can train the model only on a certain set of text. For example we can retrain the model only with a large set of astronomy academic papers, yielding us academic language. We might obtain: “The sun emits radiation.” This is the first best, but it can be very expensive: several thousand dollars at the minimum for subpar results, though techniques like LoRA (Low-Rank Adaptation of LLMs) suggest a cheaper method for some use cases. Alternatively, we can just ask for a specific tone. For example: “answer using only internal corporate documents.”
- Keep in mind: LLMs do not understand your question, they only predict your next word in the context of your question. Because of this, you can’t ask it how to better control its own output, since there has been no text on the topic.

3 Main use cases, with examples

- GPT will not be able to process a question longer than $\sim 2,500$ words. There is a memory limit. When using OpenAI’s GPT3.5 API (to Python or whichever program), you pay per “token” (~ 0.75 words). 1\$ will give 500k words. GPT4 will get 15-20x more expensive. Similar systems from Anthropic or Google are priced roughly the same as GPT3.5. If you have a powerful GPU or a cloud GPU account, you can run open-source LLMs, but at present they are inferior.
- The main uses for economists are:
 1. Cleaning data
 2. Programming and making graphs
 3. Spelling and grammar checks
 4. Summarizing literature
- Below we go through four examples of how you can use LLMs in your research.

3.1 Entity recognition: interpreting citations

Suppose we had to identify why a given scientific citation was used in an academic paper. Was the citation referring to background knowledge? Was it a use-case for an invention? Was the cited tool or technique used to make a new invention? For example, suppose we wanted to identify why Barak et. al (1997) was cited below.

β arr2-GFP trafficking was visualized in transfected HEK-293 cells on a heated (37° C.) microscope stage as described in Barak, L. S., Ferguson, S. S., Zhang, J., and Caron, M. G. (1997) *J. Biol. Chem.*, 272:27497–27500. Images were collected sequentially using single line excitation (488 nm) with a Zeiss laser scanning confocal microscope (LSM-510). For experiments assessing β arr2-GFP trafficking after agonist removal, cells were washed as described above to remove agonist and returned to a 37° C. incubator for 60 min. Colocalization of β arr2-GFP with rhodamine-labeled receptors was performed on transfected cells pre-incubated in serum-free medium containing a rhodamine-conjugated anti-HA 12CA5 mouse monoclonal antibody (1:100) for 45 min. at 37° C. Cells were then washed three times with serum-free medium, treated with the appropriate agonist at 37° C. for 30 min, and imaged by confocal microscopy. β arr2-GFP and rhodamine-labeled receptor fluorescence were performed using dual excitation (488, 568 nm) and emission (515–540 nm, GFP; 590–610 nm, rhodamine) filter sets.

To do so, we can write the following Python code, which will run after importing the OpenAI library and loading your API key (the key is how they charge you).

```
instructions = "Consider this sample from a patent. We want to know why the
stated paper was cited by this inventor. Potential reasons are
as general background knowledge, as a tool or technique to help
create the invention as an example of a use case of the
invention. State first 'Background knowledge', 'Tool/technique'
or 'Example use', then state why you think the stated article
was cited by this inventor"

patent = "Article in question: Barak et al. Patent text: TEXT DISPLAYED ABOVE"
send_to_gpt.append({"role":"system","content":instructions})
send_to_gpt.append({"role":"user","content":patent})
response=openai.ChatCompletion.create(
    model="gpt-3.5-turbo",
    messages=send_to_gpt,
    temperature=0
)
```

The temperature = 0 means that we will get fully deterministic answers, hence taking away of all of the creativity of the model, which is better for our research purposes (the temperature on the online chat is not zero). We can also try to improve the prompt by providing few-shot examples, such that I include some examples of other citations and how I would like the model to respond (e.g. *Examples:* \n *User: Article in question is Drews. Patent text: Receptors, including GPCRs, have historically been targets for drug discovery and therapeutic agents because they bind ligands, hormones, and drugs with high specificity. Approximately fifty percent of the therapeutic drugs in use today target or interact directly with GPCRs. See eg., Jurgen Drews, (2000) Drug Discovery: A Historical Perspective, Science 287:1960–1964.* \n *Response: Background knowledge*).

Running this will require 495 tokens. Now, there may be many different ways to prompt the LLM to perform this task. We could also ask in the instructions:

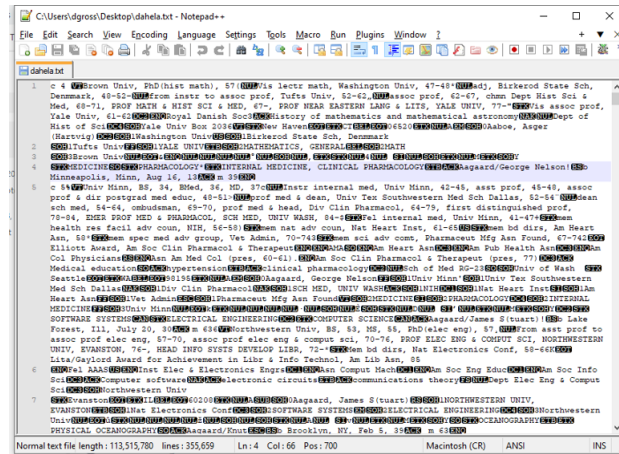
User wants to know why the stated article was cited by this inventor. Potential reasons are as general background knowledge, as a tool or technique to help create the invention, as an example of a use case of the invention. State first 'Background knowledge', 'Tool/technique' or 'Example use', then state why you think the stated article was cited by this inventor.

However, minor changes in language can result in large differences in the system's behavior. To choose the best prompt amongst several, we might implement an **ensemble approach** (a common technique in neural networks and

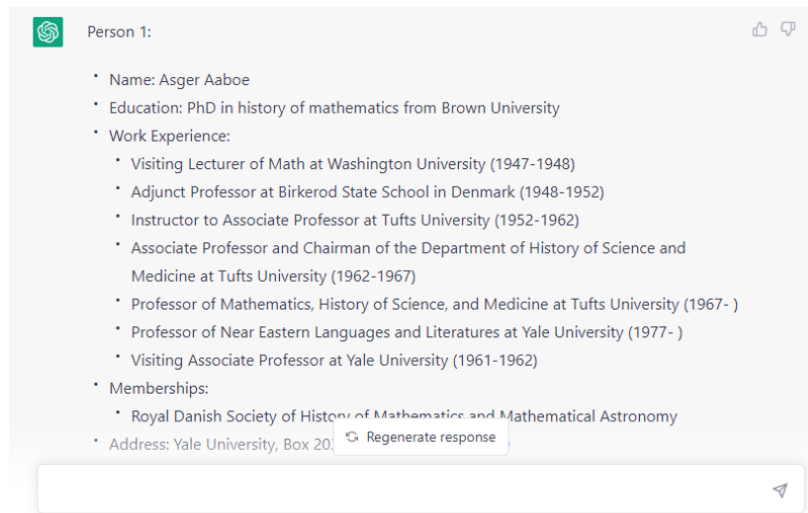
machine learning), where we ask the LLM to run all prompts and choose the most common answer amongst all prompts for each case.

3.2 Cleaning data

Supposed we obtained some data that is an old and illegible format. Opening it in .txt format might yield:



We can ask the LLM to figure out the structure of the data and clean it to obtain:



ChatGPT Feb 13 Version. Free Research Preview. Our goal is to make AI systems more natural and safe to interact with. Your feedback will help us improve.

3.3 Cleaning text

We have all struggled to OCR or to copy and paste a text with formulas or mathematical symbols. If we tried to OCR the following paragraph of a paper (from Bryan (2020)), we would obtain the messy and typo-ridden result on the right.

Original PDF

Consider first three results which are either known in the literature, or straightforward implications of known results.

1) If p_i are *independent* and search is *simultaneous*, optimal search opens all boxes p_j such that $\prod_{i:i < j} (1 - p_i) p_j R \geq c$, where boxes are ordered by size. We therefore find the prize with probability $1 - \prod_{i \leq j^*} (1 - p_i)$ where j^* is the smallest box j such that $\prod_{i:i < j} (1 - p_i) p_j R \geq c$. This is a direct implication of the marginal improvement algorithm in Chade and Smith [2006].⁶

2) If p_i are *independent* and search is *sequential*, optimal search opens boxes in order of p_i until the prize is found, or until there exists no box such that $p_i R \geq c$. We therefore find the prize with probability $1 - \prod_{i:p_i R \geq c} (1 - p_i)$ (Weitzman [1979]).⁷ Note that sequential search opens weakly more boxes than simultaneous search.

OCR result

Consider first three results which are either known in the literature, or straightforward implications of known results.

1) If p_i are independent and search is simultaneous, optimal search opens all

boxes p_j such that Q

$\prod_{i:i < j} (1 - p_i) p_j R \geq c$, where boxes are ordered by size. We therefore find the prize with probability $1 - \prod_{i \leq j^*} (1 - p_i)$

where j^* is the smallest box j such

that $Q \prod_{i:i < j} (1 - p_i) p_j R \geq c$. This is a direct implication of the marginal improvement

algorithm in Chade and Smith [2006].⁶

2) If p_i are independent and search is sequential, optimal search opens boxes in

order of p_i until the prize is found, or until there exists no box such that $p_i R \geq c$. We

therefore find the prize with probability $1 -$

$\prod_{i:p_i R \geq c} (1 - p_i)$ (Weitzman [1979]).⁷ Note

that sequential search opens weakly more boxes than simultaneous search.

We can ask the LLM to clean this and put it in Latex format. After providing it with the context of the paper where it is coming from, we obtain.

ChatGPT result

Consider first three results which are either known in the literature, or straightforward implications of known results.

If p_i are independent and search is simultaneous, optimal search opens all boxes p_j such that $\prod_{i:i < j} (1 - p_i) p_j R \geq c$, where boxes are ordered by size. We therefore find the prize with probability $1 - \prod_{i \leq j^*} (1 - p_i)$ where j^* is the smallest box j such that $\prod_{i:i < j} (1 - p_i) p_j R \geq c$. This is a direct implication of the marginal improvement algorithm in Chade and Smith [2006].

If p_i are independent and search is sequential, optimal search opens boxes in order of p_i until the prize is found, or until there exists no box such that $p_i R \geq c$. We therefore find the prize with probability $1 - \prod_{i:p_i R \geq c} (1 - p_i)$ (Weitzman [1979]). Note that sequential search opens weakly more boxes than simultaneous search.

ChatGPT result (in Latex)

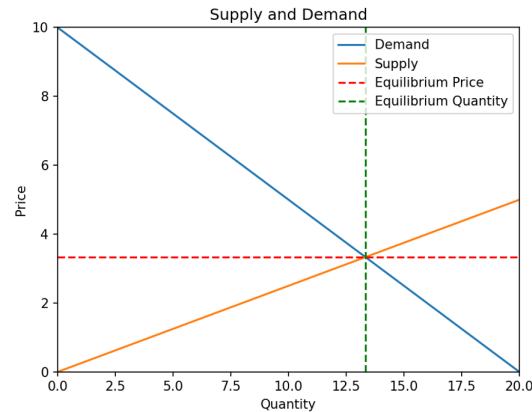
Consider first three results which are either known in the literature, or straightforward implications of known results. If p_i are independent and search is simultaneous, optimal search opens all boxes p_j such that $\prod_{i:i < j} (1 - p_i) p_j R \geq c$, where boxes are ordered by size. We therefore find the prize with probability $1 - \prod_{i \leq j^*} (1 - p_i)$ where j^* is the smallest box j such that $\prod_{i:i < j} (1 - p_i) p_j R \geq c$. This is a direct implication of the marginal improvement algorithm in Chade and Smith [2006]. If p_i are independent and search is sequential, optimal search opens boxes in order of p_i until the prize is found, or until there exists no box such that $p_i R \geq c$. We therefore find the prize with probability $1 - \prod_{i:p_i R \geq c} (1 - p_i)$ (Weitzman [1979]). Note that sequential search opens weakly more boxes than simultaneous search.

3.4 Writing code / making graphs

It is well known that ChatGPT is an effective tool to improve coding. Here we show a simple example to illustrate how it can be used to make charts. Suppose we asked the model:

Give me the simplest python code for a chart of supply and demand, $Q=20-2P$, $Q=4P$, labeling supply, demand, the axes, total surplus under perfect competition and total profit under perfect competition. First write out the equations you will graph, thinking step by step, then create the Latex code.

As is usual, the output from codes requires some adjustments. In our case, we asked it to make sure that quantity was on the x-axis and price on the y-axis, and to adjust the axis bounds to obtain: (we could have also requested any other color or formatting changes)



3.5 Summarization

With LLMs, we can build tools to query large databases of text with the model, for example to obtain a summary of one of the documents in the database. I did so for a large set of academic papers to be able to do the following.

Give the sample size and main methodology of the Bryan et al startup hiring paper

The Bryan et al paper on startup hiring examines the impact of information frictions on employee sorting among startups. The study consists of two randomized controlled trials (RCTs) conducted in collaboration with a science-based entrepreneurship program (SEP) that works with high-potential startups.

The first RCT involved 26 participating startups that advertised job openings via a job board created for the study. Treatments were assigned at the applicant level, randomly changing the visibility of two aspects of information about each firm: a control, information about business model quality, information about science quality, or both. The study tracked the number of applications submitted to each firm.

This requires scanning each paper into an embedded vector database. Vector embeddings are a method from natural language processing which encodes text as a high-dimensional vector, allowing very simple mathematical techniques to be performed. For instance, the similarity of two blocks of embedded text can be found by taking their dot product. By embedding a user question, or a modified version of their question, and comparing it with different pre-prepared chunks of text (say, 200 words each, created in an overlapping fashion) using dot products, we can quickly find the text most likely to contain the answer. This allows us to only send a small amount of “possibly relevant” text to the LLM, and hence avoid any memory limits. In the [Virtual TA](#) guide I mentioned above, I write a code to embed the documents for a class, but you can also review [OpenAI’s guide](#).

4 Some tips and tricks to conclude

- When feeding the model data, use few-shot prompting: provide the model with as many examples as you can of what you need to be done.
- Play with different prompts: use an “ensemble approach” to find the optimal prompt to ask the model. Keep in mind optimal prompting is not well understood yet.
- There are many methods to avoid hallucinations. One is to use an ensemble approach and only treat an answer as true if it was generated by the majority of slightly modified prompts. Another is to turn the temperature down to zero - these answers will be less creative and hence have less hallucination. A third is to explicitly ask the model

questions. For example, after giving an LLM context and getting a response to a question, pass the response and the original question back to the LLM, and ask with the prompt "Is the response XXX stated in the attached context?" I find GPT-4 works very well for this type of follow-up

- Do not use LLMs to do math. Suppose you need to multiply two 8-digit numbers together. That computation has never been done before, so the LLM will give you the wrong answer.
- To handle the model's memory limit, we can (1) break the content to be processed into different chunks, (2) use vector embedding and cosine similarity to find the relevant text for a given problem
- Soon, (1) OCR will be replaced by LLM-based reading of images, (2) the LLM will run code itself, (3) LLMs will be able to search the web to verify facts.

References

Bryan, K. A. (2020), 'Sequential search on a partition', *Working Paper* .

URL: <https://www.kevinbryanecon.com/BryanPartitionSearch.pdf>

Kalliamvakou, E. (2022), 'Quantifying github copilot's impact on developer productivity and happiness', *Github* .

URL: <https://github.blog/2022-09-07-research-quantifying-github-copilots-impact-on-developer-productivity-and-happiness/>

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2017), 'Attention is all you need', *arXiv preprint arXiv:1706.03762* .

URL: <https://doi.org/10.48550/arXiv.1706.03762>