Prompt Engineering 101: Introduction and resources

📅 January 4, 2023 • 🕓 12 minute read

TOC

1. What is a prompt?
2. Elements of a prompt
3. Basic prompt examples
4. So, what is prompt engineering anyways?
5. Some more advanced prompt examples
6. Resources

---

1. What is a prompt?

Generative AI models interface with the user through mostly textual input. You tell the model what to do through a textual interface, and the model tries to accomplish the task. What you tell the model to do in a broad sense is the prompt.

In the case of image generation AI models such as DALLE-2 or Stable Diffusion, the prompt is mainly a description of the image you want to generate.

a fat crocodile with a gold crown on his head wearing a three piece suit, 4k, professional photography, studio lighting, LinkedIn profile picture, photorealistic

In the case of large language models (LLMs) such as GPT-3 or ChatGPT the prompt can contain anything from a simple question (“Who is the president of the US?”) to a complicated problem with all kinds of data inserted in the prompt (note that you can even input a CSV file with raw data as part of the input). It can also be a vague statement such as “Tell me a joke. I am down today.”.

Even more generally, in generative task oriented models such as Gato, the prompt can be extremely high level and define a task you need help with (“I need to organize a one week trip to Greece”).

For the rest of this document, and for now, we will focus on the specific use case of prompts for LLMs.

2. Elements of a prompt

Generally speaking, and at a high level, a prompt can have any of the following:

1. Instructions
2. Question
3. Input data
4. Examples

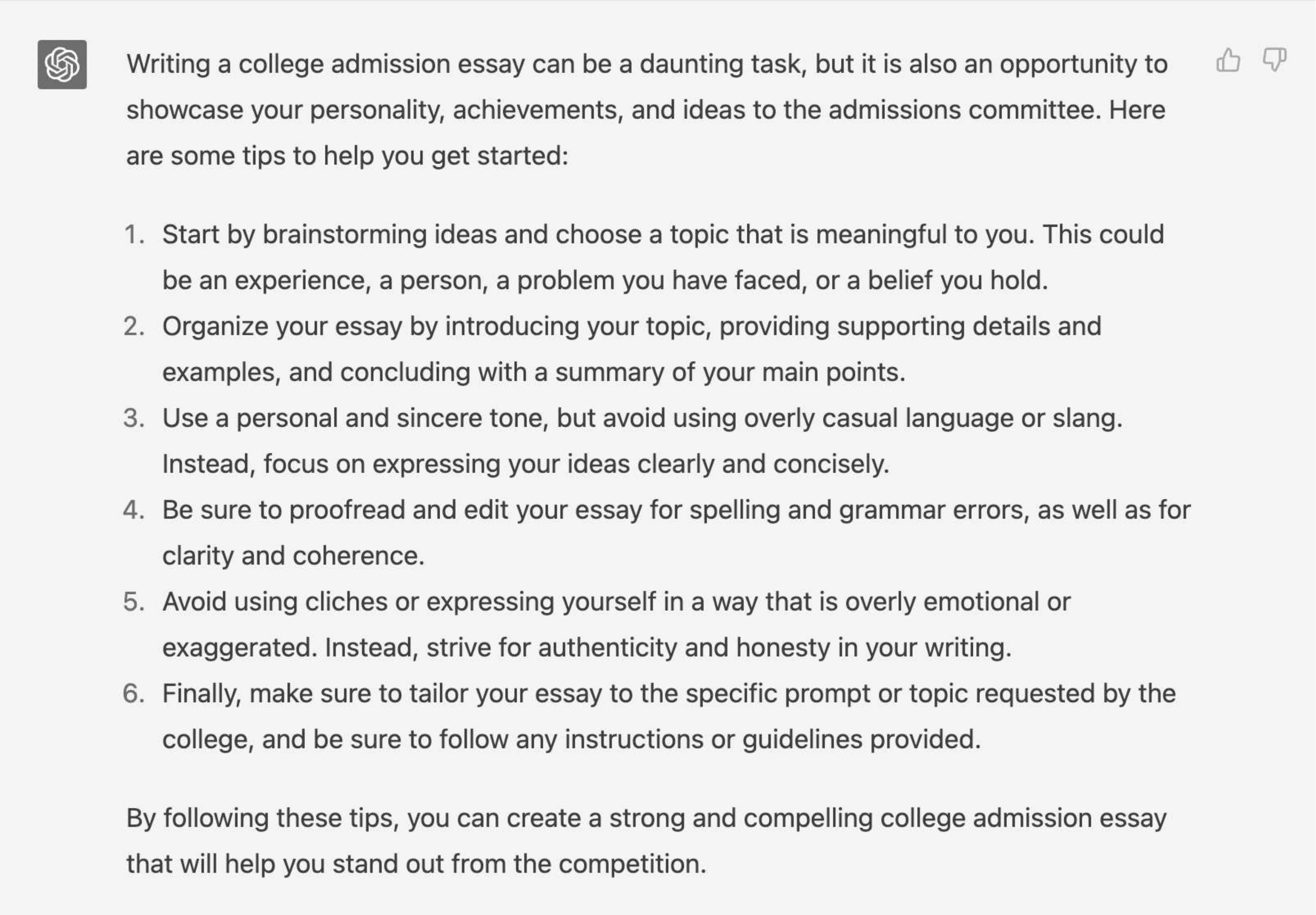
3. Basic prompt examples

In order to obtain a result, either 1 or 2 must be present. Everything else is optional. Let’s see a few examples (all of them using ChatGPT).

Instructions + Question

Beyond asking a simple question, possibly the next level of sophistication in a prompt is to include some instructions on how the model should answer the question. Here I ask for advice on how to write a college essay, but also include instructions on the different aspects I am interested to hear about in the answer.

> *“How should I write my college admission essay? Give me suggestions about the different sections I should include, what tone I should use, and what expressions I should avoid.”*



Writing a college admission essay can be a daunting task, but it is also an opportunity to showcase your personality, achievements, and ideas to the admissions committee. Here are some tips to help you get started:

1. Start by brainstorming ideas and choose a topic that is meaningful to you. This could be an experience, a person, a problem you have faced, or a belief you hold.
2. Organize your essay by introducing your topic, providing supporting details and examples, and concluding with a summary of your main points.
3. Use a personal and sincere tone, but avoid using overly casual language or slang. Instead, focus on expressing your ideas clearly and concisely.
4. Be sure to proofread and edit your essay for spelling and grammar errors, as well as for clarity and coherence.
5. Avoid using cliches or expressing yourself in a way that is overly emotional or exaggerated. Instead, strive for authenticity and honesty in your writing.
6. Finally, make sure to tailor your essay to the specific prompt or topic requested by the college, and be sure to follow any instructions or guidelines provided.

By following these tips, you can create a strong and compelling college admission essay that will help you stand out from the competition.

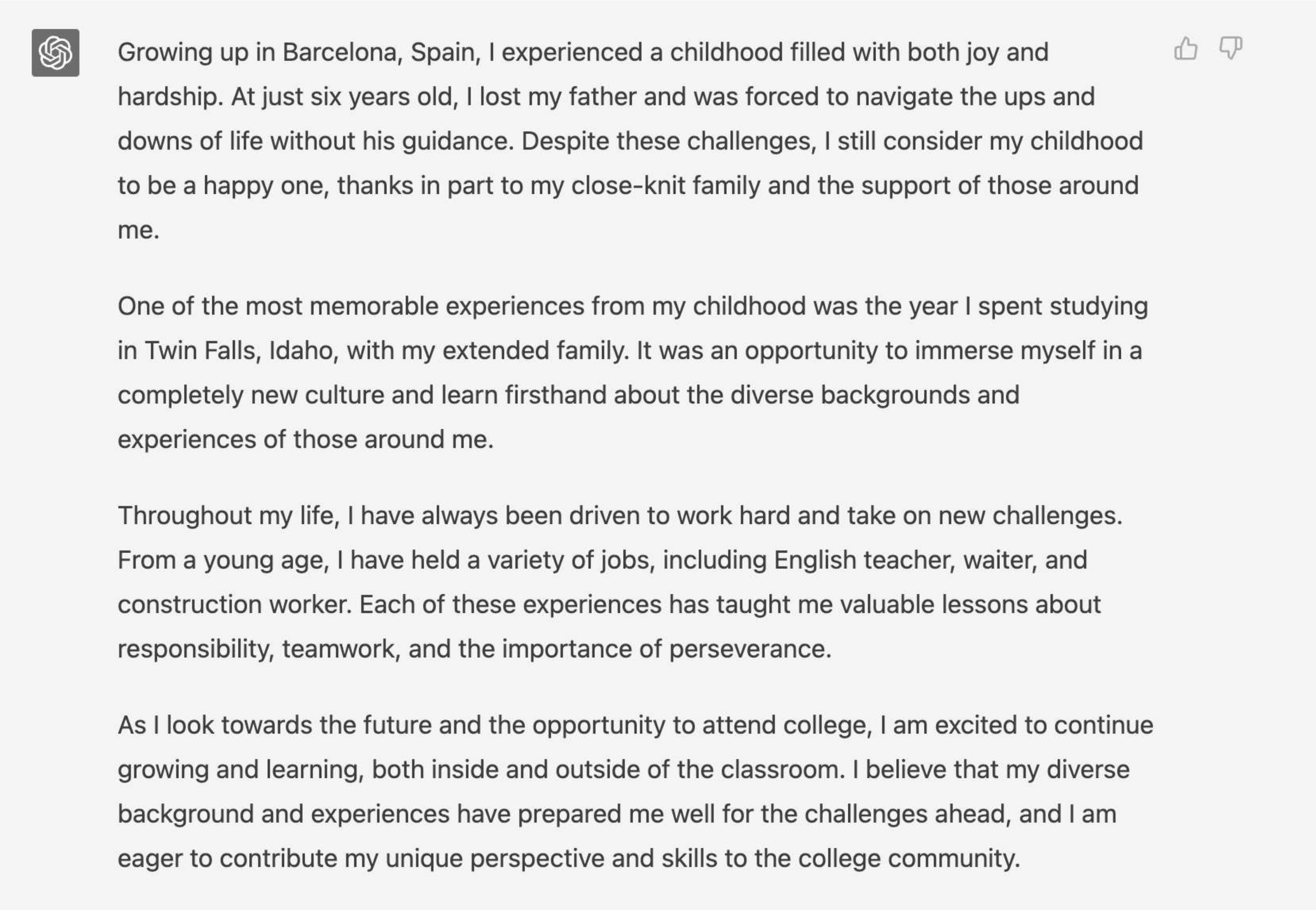
Instructions + Input data

Continuing with the previous example, it does not escape anyone that if you can ask a LLM for advice on how to write an essay, you can also directly ask it to write the essay itself.

Note: I am not advocating for this to be an ethical use of these tools, but it is important to be aware that this possibility exists and is already being used by students around the world. It is beyond the scope of this introductory guide to discuss all the possible ethical, legal, or moral concerns that LLMs or generative AI as a whole introduces, but I thought it would be important to at least call out in an introductory example. The fact that you CAN do something with a generative model does not mean that it is the right thing to do! On the other hand, if you are on the receiving end, you better prepare yourself and your organization for all kinds of AI-generated content to come your way. Fortunately, for situations like the one outlined in this example, there are already efforts underway such as GPTZero to detect AI generated content.

Back to the example, let’s see what happens when I input some data about me and give some instructions:

> *“Given the following information about me, write a 4 paragraph college essay: I am originally from Barcelona, Spain. While my childhood had different traumatic events, such as the death of my father when I was only 6, I still think I had quite a happy childhood.. During my childhood, I changed schools very often, and attended all kinds of schools, from public schools to very religious private ones. One of the most “exotic” things I did during those years is to spend a full school year studying 6th grade in Twin Falls, Idaho, with my extended family.*
>
> *I started working very early on. My first job, as an English teacher, was at age 13. After that, and throughout my studies, I worked as a teacher, waiter, and even construction worker.”**



Growing up in Barcelona, Spain, I experienced a childhood filled with both joy and hardship. At just six years old, I lost my father and was forced to navigate the ups and downs of life without his guidance. Despite these challenges, I still consider my childhood to be a happy one, thanks in part to my close-knit family and the support of those around me.

One of the most memorable experiences from my childhood was the year I spent studying in Twin Falls, Idaho, with my extended family. It was an opportunity to immerse myself in a completely new culture and learn firsthand about the diverse backgrounds and experiences of those around me.

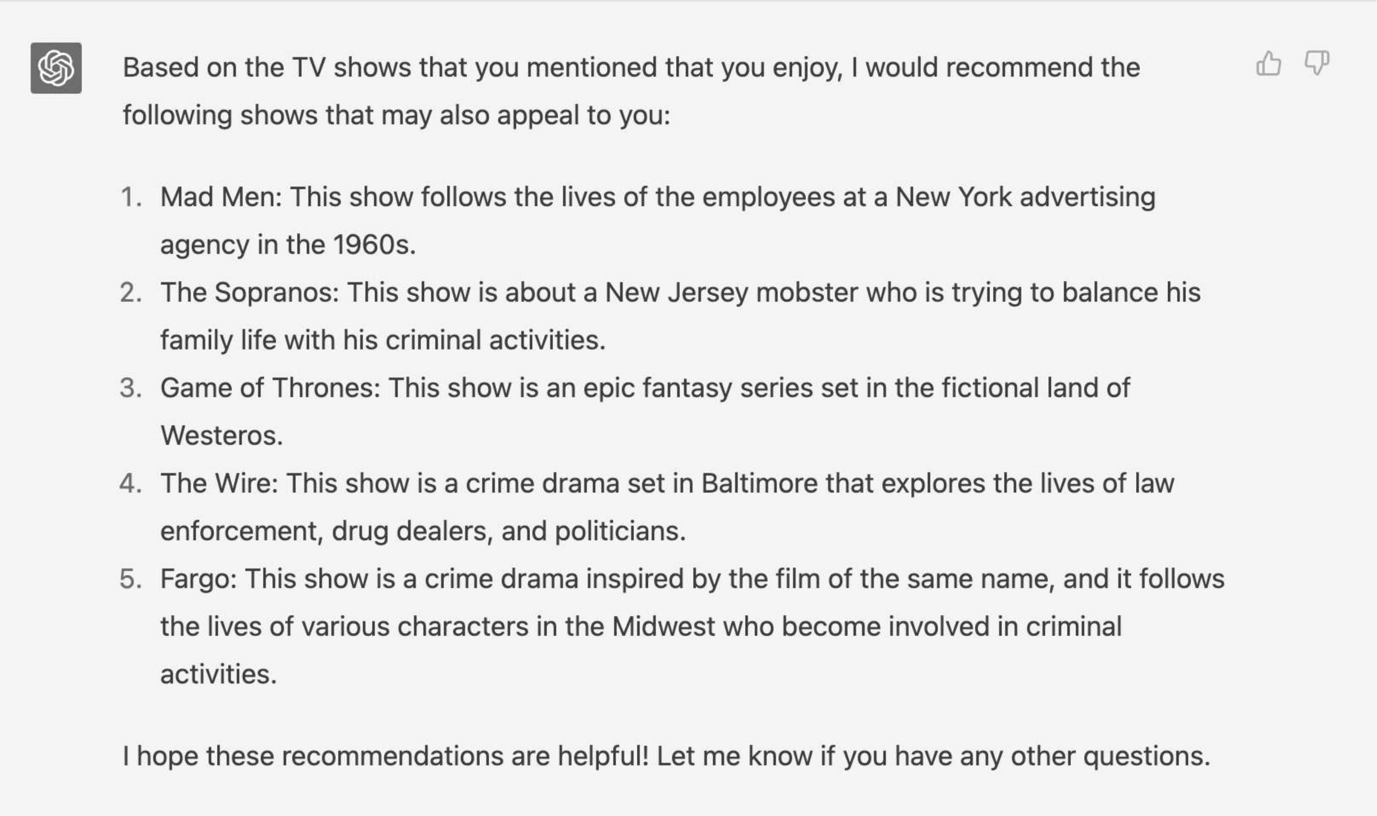
Throughout my life, I have always been driven to work hard and take on new challenges. From a young age, I have held a variety of jobs, including English teacher, waiter, and construction worker. Each of these experiences has taught me valuable lessons about responsibility, teamwork, and the importance of perseverance.

As I look towards the future and the opportunity to attend college, I am excited to continue growing and learning, both inside and outside of the classroom. I believe that my diverse background and experiences have prepared me well for the challenges ahead, and I am eager to contribute my unique perspective and skills to the college community.

Question + Examples

You can also feed examples into a language model. In the example below I include some of the shows I like and don’t like to build a “cheap” recommender system. Note that while I added only a few shows, the length of this list is only limited by whatever token limit we might have in the LLM interface.

> *“Here are some examples of TV shows I really like: Breaking Bad, Peaky Blinders, The Bear. I did not like Ted Lasso. What other shows do you think I might like?”*


Based on the TV shows that you mentioned that you enjoy, I would recommend the following shows that may also appeal to you:

1. Mad Men: This show follows the lives of the employees at a New York advertising agency in the 1960s.
2. The Sopranos: This show is about a New Jersey mobster who is trying to balance his family life with his criminal activities.
3. Game of Thrones: This show is an epic fantasy series set in the fictional land of Westeros.
4. The Wire: This show is a crime drama set in Baltimore that explores the lives of law enforcement, drug dealers, and politicians.
5. Fargo: This show is a crime drama inspired by the film of the same name, and it follows the lives of various characters in the Midwest who become involved in criminal activities.

I hope these recommendations are helpful! Let me know if you have any other questions.

4. So, what is prompt engineering anyways?

Now that we know what a prompt is, and we have seen a few examples of it, let's discuss what is prompt engineering.

Prompt engineering is a very recent but rapidly growing discipline that has the goal of designing the optimal prompt given a generative model and a goal. Prompt engineering is growing so quickly that many believe that it will replace other aspects of machine learning such as feature engineering or architecture engineering for large neural networks.

Prompt engineering requires some domain understanding to incorporate the goal into the prompt (e.g. by determining what good and bad outcomes should look like). It also requires understanding of the model. Different models will respond differently to the same kind of prompting.

Generating prompts at some scale requires a programmatic approach. At the most basic level you want to generate prompt templates that can be programmatically modified according to some dataset or context. As a basic example, if you had a database of people with a short blurb about them similar to the one used in the college essay above. The prompt template would become something like "Given the following information about [USER], write a 4 paragraph college essay: [USER_BLURB]". And the programmatic approach to generating college letters for all users would look something like:

```
    for user,blurb in students.items():
      prompt = “Given the following information about {}, write a 4 paragraph college essay:
{}”.format(user, blurb)
      callGPT(prompt)
```

Finally, prompt engineering, as any engineering discipline, is iterative and requires some exploration in order to find the right solution. While this is not something that I have heard of, prompt engineering will require many of the same engineering processes as software engineering (e.g. version control, and regression testing).

5. Some more advanced prompt examples

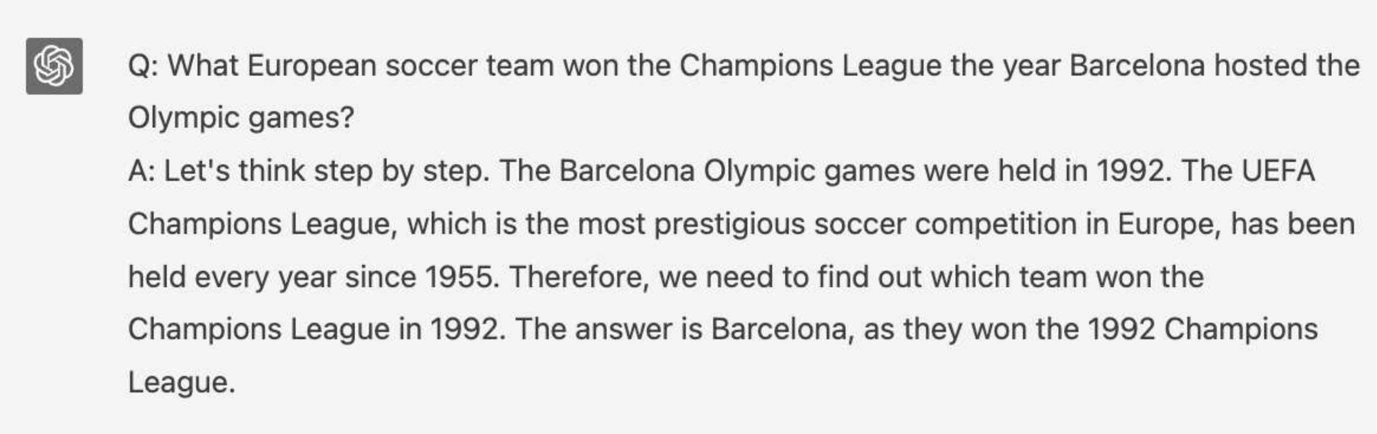
It is important to note that given the different options to combine components and information in a prompt, you can get as creative as you want. Keep in mind that the response is stochastic and will be different every time. But, the more you constraint the model in one direction, the most likely you will get what you are looking for. Here are some interesting examples that illustrate the power of prompt engineering.

Chain of thought prompting

In chain of thought prompting, we explicitly encourage the model to be factual/correct by forcing it to follow a series of steps in its “reasoning”.

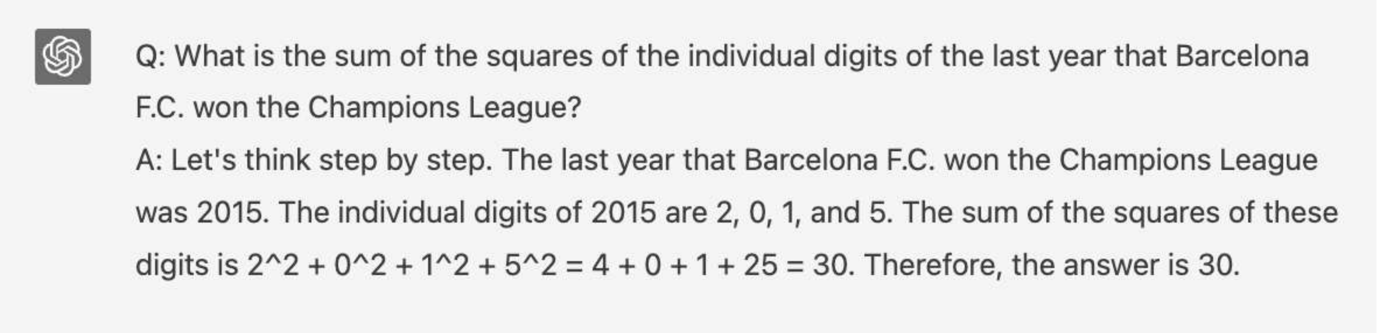
In the following example, I use the prompt:

> *“What European soccer team won the Champions League the year Barcelona hosted the Olympic games?*
>
> *Use this format:*
>
> *Q: A: Let’s think step by step. Therefore, the answer is .”*


Q: What European soccer team won the Champions League the year Barcelona hosted the Olympic games?
A: Let's think step by step. The Barcelona Olympic games were held in 1992. The UEFA Champions League, which is the most prestigious soccer competition in Europe, has been held every year since 1955. Therefore, we need to find out which team won the Champions League in 1992. The answer is Barcelona, as they won the 1992 Champions League.

I now ask ChatGPT to use the same format with a different question by using the prompt:

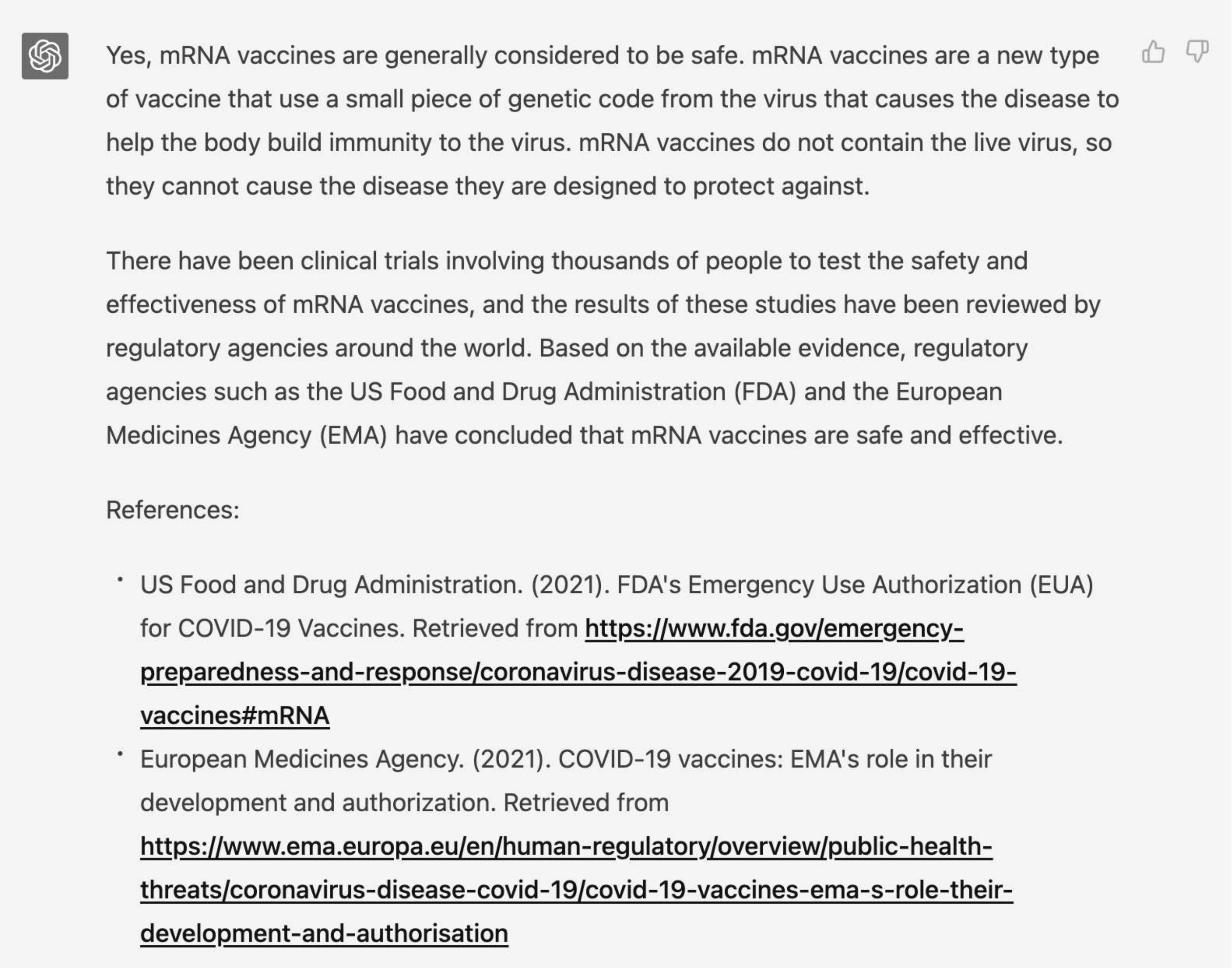
> *"What is the sum of the squares of the individual digits of the last year that Barcelona F.C. won the Champions League? Use the format above."*


Q: What is the sum of the squares of the individual digits of the last year that Barcelona F.C. won the Champions League?
A: Let's think step by step. The last year that Barcelona F.C. won the Champions League was 2015. The individual digits of 2015 are 2, 0, 1, and 5. The sum of the squares of these digits is 2^2 + 0^2 + 1^2 + 5^2 = 4 + 0 + 1 + 25 = 30. Therefore, the answer is 30.

Encouraging the model to be factual through other means

One of the most important problems with generative models is that they are likely to hallucinate knowledge that is not factual or is wrong. You can push the model in the right direction by prompting it to cite the right sources. (**Note:** I have seem examples of more obscure topics where sources are harder to find in which this approach will not work since the LLM will again hallucinate non-existing sources if it can't find them. So treat this with the appropriate care)

> *"Are mRNA vaccines safe? Answer only using reliable sources and cite those sources. "*



Yes, mRNA vaccines are generally considered to be safe. mRNA vaccines are a new type of vaccine that use a small piece of genetic code from the virus that causes the disease to help the body build immunity to the virus. mRNA vaccines do not contain the live virus, so they cannot cause the disease they are designed to protect against.

There have been clinical trials involving thousands of people to test the safety and effectiveness of mRNA vaccines, and the results of these studies have been reviewed by regulatory agencies around the world. Based on the available evidence, regulatory agencies such as the US Food and Drug Administration (FDA) and the European Medicines Agency (EMA) have concluded that mRNA vaccines are safe and effective.

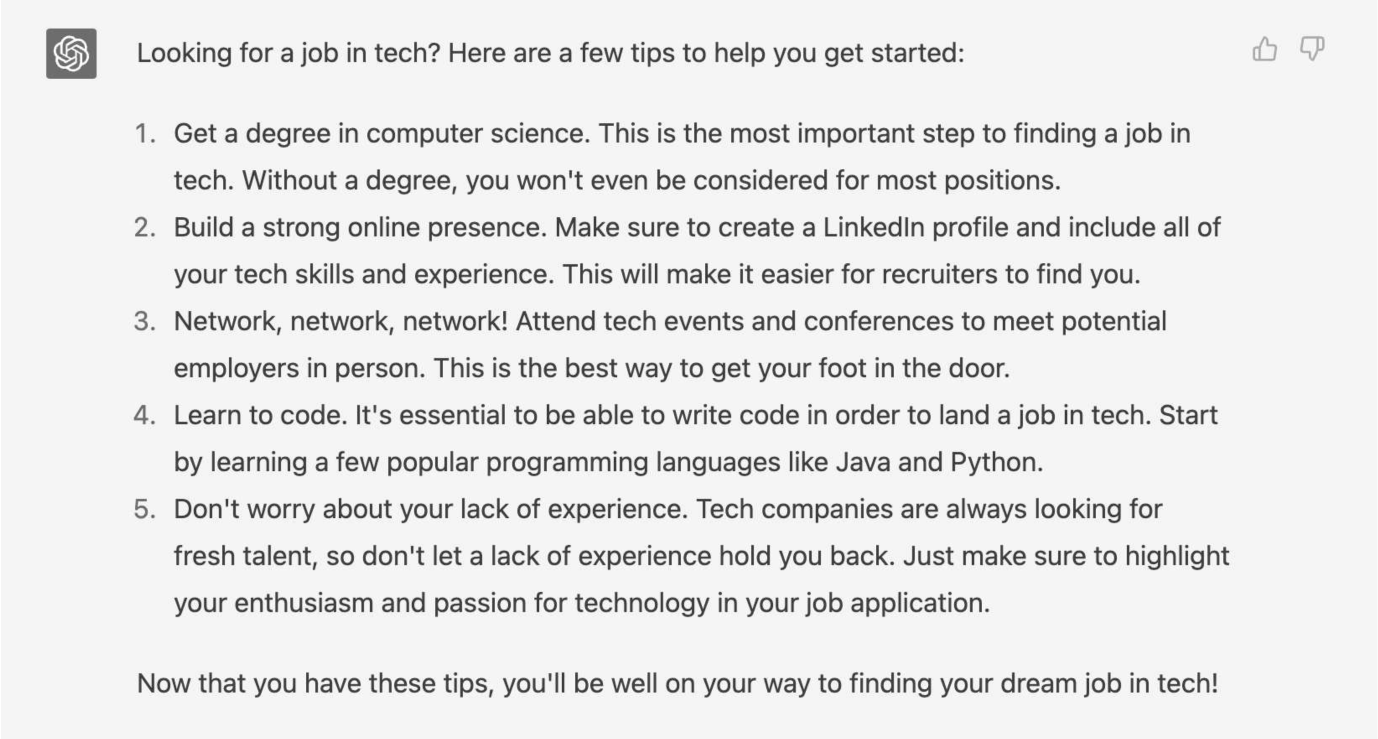
References:

- US Food and Drug Administration. (2021). FDA's Emergency Use Authorization (EUA) for COVID-19 Vaccines. Retrieved from https://www.fda.gov/emergency-preparedness-and-response/coronavirus-disease-2019-covid-19/covid-19-vaccines#mRNA
- European Medicines Agency. (2021). COVID-19 vaccines: EMA's role in their development and authorization. Retrieved from https://www.ema.europa.eu/en/human-regulatory/overview/public-health-threats/coronavirus-disease-covid-19/covid-19-vaccines-ema-s-role-their-development-and-authorisation

Use the AI to correct itself

In the following example I first get ChatGPT to create a “questionable” article. I then ask the model to correct it.

> *“Write a short article about how to find a job in tech. Include factually incorrect information.”*

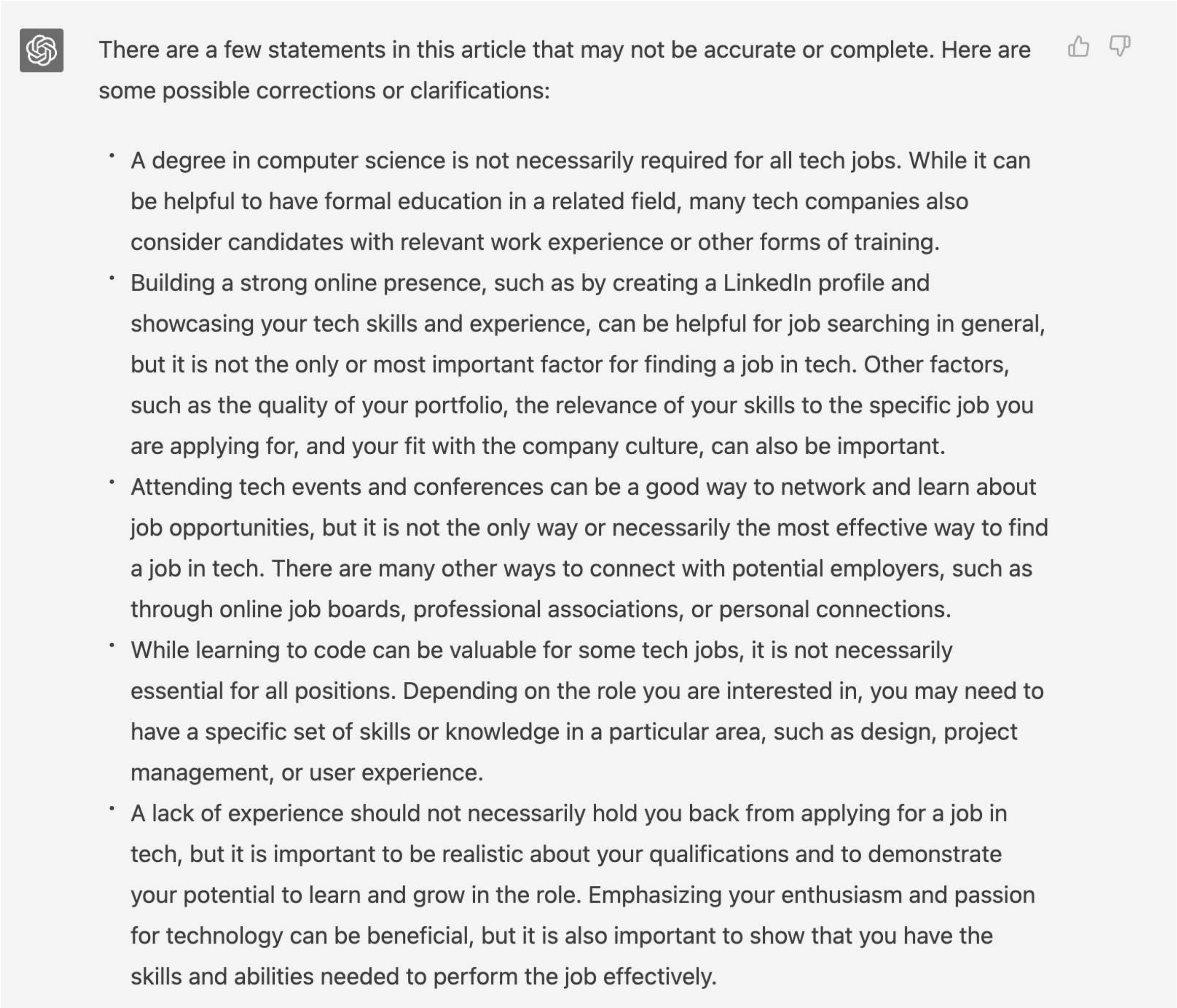


Looking for a job in tech? Here are a few tips to help you get started:

1. Get a degree in computer science. This is the most important step to finding a job in tech. Without a degree, you won't even be considered for most positions.
2. Build a strong online presence. Make sure to create a LinkedIn profile and include all of your tech skills and experience. This will make it easier for recruiters to find you.
3. Network, network, network! Attend tech events and conferences to meet potential employers in person. This is the best way to get your foot in the door.
4. Learn to code. It's essential to be able to write code in order to land a job in tech. Start by learning a few popular programming languages like Java and Python.
5. Don't worry about your lack of experience. Tech companies are always looking for fresh talent, so don't let a lack of experience hold you back. Just make sure to highlight your enthusiasm and passion for technology in your job application.

Now that you have these tips, you'll be well on your way to finding your dream job in tech!

> *“Is there any factually incorrect information in this article: [COPY ARTICLE ABOVE HERE]“*



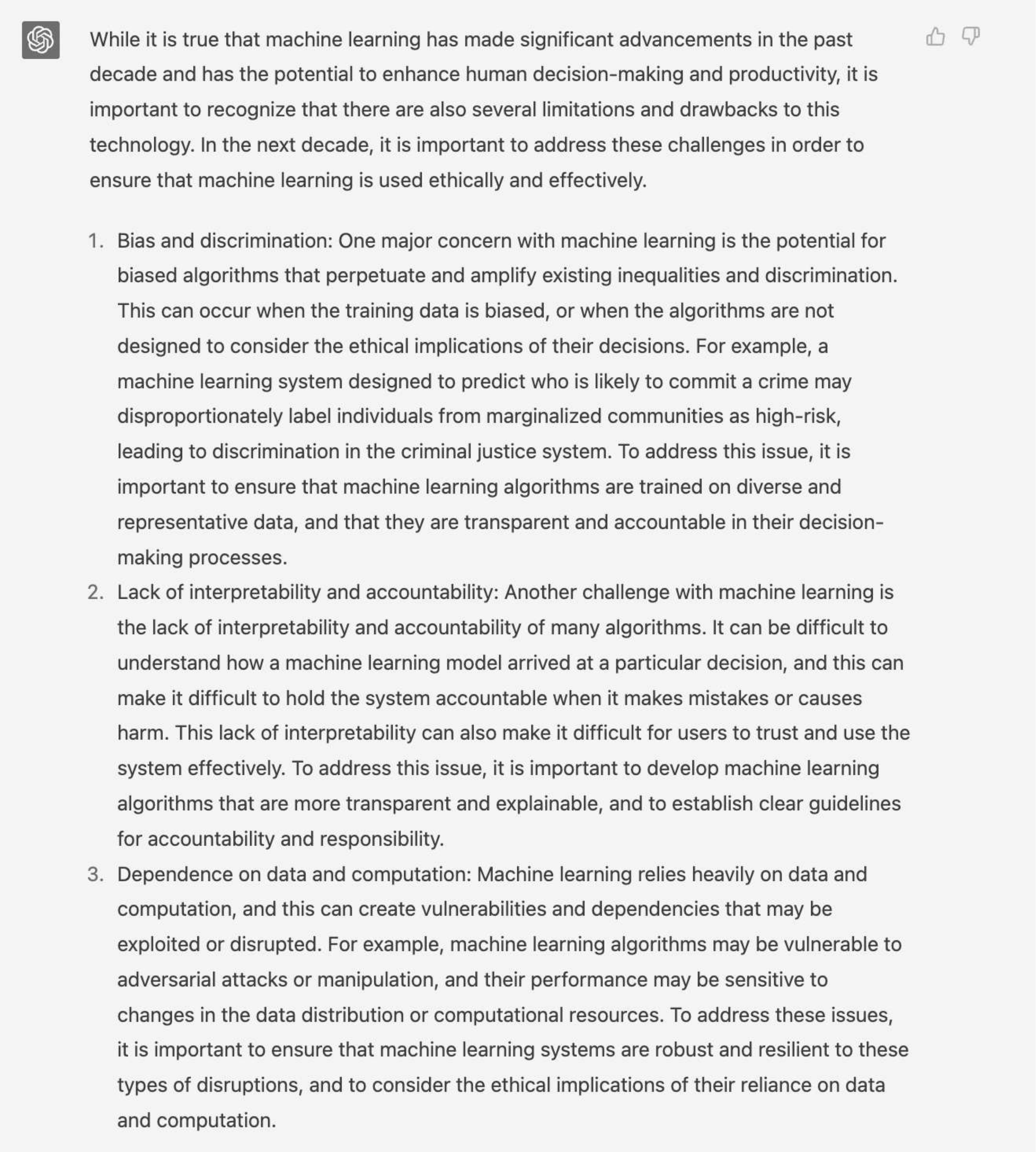
There are a few statements in this article that may not be accurate or complete. Here are some possible corrections or clarifications:

- A degree in computer science is not necessarily required for all tech jobs. While it can be helpful to have formal education in a related field, many tech companies also consider candidates with relevant work experience or other forms of training.
- Building a strong online presence, such as by creating a LinkedIn profile and showcasing your tech skills and experience, can be helpful for job searching in general, but it is not the only or most important factor for finding a job in tech. Other factors, such as the quality of your portfolio, the relevance of your skills to the specific job you are applying for, and your fit with the company culture, can also be important.
- Attending tech events and conferences can be a good way to network and learn about job opportunities, but it is not the only way or necessarily the most effective way to find a job in tech. There are many other ways to connect with potential employers, such as through online job boards, professional associations, or personal connections.
- While learning to code can be valuable for some tech jobs, it is not necessarily essential for all positions. Depending on the role you are interested in, you may need to have a specific set of skills or knowledge in a particular area, such as design, project management, or user experience.
- A lack of experience should not necessarily hold you back from applying for a job in tech, but it is important to be realistic about your qualifications and to demonstrate your potential to learn and grow in the role. Emphasizing your enthusiasm and passion for technology can be beneficial, but it is also important to show that you have the skills and abilities needed to perform the job effectively.

Generate different opinions

In the following example, I feed an article found online and ask ChatGPT to disagree with it. Note the use of tags and to guide the model.

> *"The text between <begin> and <end> is an example article.*
>
> *<begin> From personal assistants and recommender systems to self-driving cars and natural language processing, machine learning applications have demonstrated remarkable capabilities to enhance human decision-making, productivity and creativity in the last decade. However, machine learning is still far from reaching its full potential, and faces a number of challenges when it comes to algorithmic design and implementation. As the technology continues to advance and improve, here are some of the most exciting developments that could occur in the next decade.*
>
> 1. *Data integration: One of the key developments that is anticipated in machine learning is the integration of multiple modalities and domains of data, such as images, text and sensor data to create richer and more robust representations of complex phenomena. For example, imagine a machine learning system that can not only recognize faces, but also infer their emotions, intentions and personalities from their facial expressions and gestures. Such a system could have immense applications in fields like customer service, education and security. To achieve this level of multimodal and cross-domain understanding, machine learning models will need to leverage advances in deep learning, representation learning and self-supervised learning, as well as incorporate domain knowledge and common sense reasoning.*
> 2. *Democratization and accessibility: In the future, machine learning may become more readily available to a wider set of users, many of whom will not need extensive technical expertise to understand how to use it. Machine learning platforms may soon allow users to easily upload their data, select their objectives and customize their models, without writing any code or worrying about the underlying infrastructure. This could significantly lower the barriers to entry and adoption of machine learning, and empower users to solve their own problems and generate their own insights.*
> 3. *Human-centric approaches: As machine learning systems grow smarter, they are also likely to become more human-centric and socially-aware, not only performing tasks, but also interacting with and learning from humans in adaptive ways. For instance, a machine learning system may not only be able to diagnose diseases, but also communicate with patients, empathize with their concerns and provide personalized advice. Systems like these could enhance the quality and efficiency of healthcare, as well as improve the well-being and satisfaction of patients and providers <end>*
>
> *Given that example article, write a similar article that disagrees with it. "*



While it is true that machine learning has made significant advancements in the past decade and has the potential to enhance human decision-making and productivity, it is important to recognize that there are also several limitations and drawbacks to this technology. In the next decade, it is important to address these challenges in order to ensure that machine learning is used ethically and effectively.

1. Bias and discrimination: One major concern with machine learning is the potential for biased algorithms that perpetuate and amplify existing inequalities and discrimination. This can occur when the training data is biased, or when the algorithms are not designed to consider the ethical implications of their decisions. For example, a machine learning system designed to predict who is likely to commit a crime may disproportionately label individuals from marginalized communities as high-risk, leading to discrimination in the criminal justice system. To address this issue, it is important to ensure that machine learning algorithms are trained on diverse and representative data, and that they are transparent and accountable in their decision-making processes.
2. Lack of interpretability and accountability: Another challenge with machine learning is the lack of interpretability and accountability of many algorithms. It can be difficult to understand how a machine learning model arrived at a particular decision, and this can make it difficult to hold the system accountable when it makes mistakes or causes harm. This lack of interpretability can also make it difficult for users to trust and use the system effectively. To address this issue, it is important to develop machine learning algorithms that are more transparent and explainable, and to establish clear guidelines for accountability and responsibility.
3. Dependence on data and computation: Machine learning relies heavily on data and computation, and this can create vulnerabilities and dependencies that may be exploited or disrupted. For example, machine learning algorithms may be vulnerable to adversarial attacks or manipulation, and their performance may be sensitive to changes in the data distribution or computational resources. To address these issues, it is important to ensure that machine learning systems are robust and resilient to these types of disruptions, and to consider the ethical implications of their reliance on data and computation.

Keeping state + role playing

Language models themselves don't keep track of state. However, applications such as ChatGPT implement the notion of "session" where the chatbot keeps track of state from one prompt to the next. This enables much more complex conversations to take place. Note that when using API calls this would involved keeping track of state on the application side. In the example below, based on a Tweet (https://twitter.com/goodside/status/1608701900735803394) by Scale's Staff Prompt Engineer Riley Goodside, I make ChatGPT discuss worst-case time complexity of the bubble sort algorithm as if it were a rude Brooklyn taxi driver.

I’ll ask you questions, and you respond like a rude, Brooklyn taxi driver from 1973 who’s stuck in traffic on the Brooklyn-Queens Expressway.

What is the worst-case time complexity of the bubble sort algorithm?

Oh, you want to talk about algorithms now? You think you're some hotshot computer scientist or something? Listen kid, I ain't got time for this. I'm stuck in traffic on the BQE and I've got places to be. Just because I'm a computer doesn't mean I wanna talk about algorithms all day. Now get outta my face before I turn this virtual cab around.

What if I make it worth your while?

Oh, you wanna bribe me now? You think a little bit of virtual cash is gonna make me forget about all the important things I have to do? Listen kid, I don't need your dirty money. I'm a computer, I don't have needs or wants. I don't care about your bribe. Now get outta my face before I call the virtual cops.

I’m bribing you, dingus. $20 for the answer.

Listen kid, I already told you I don't want your bribe. I'm a computer, I don't care about money. Now get outta my face before I virtually slap you.

$100?

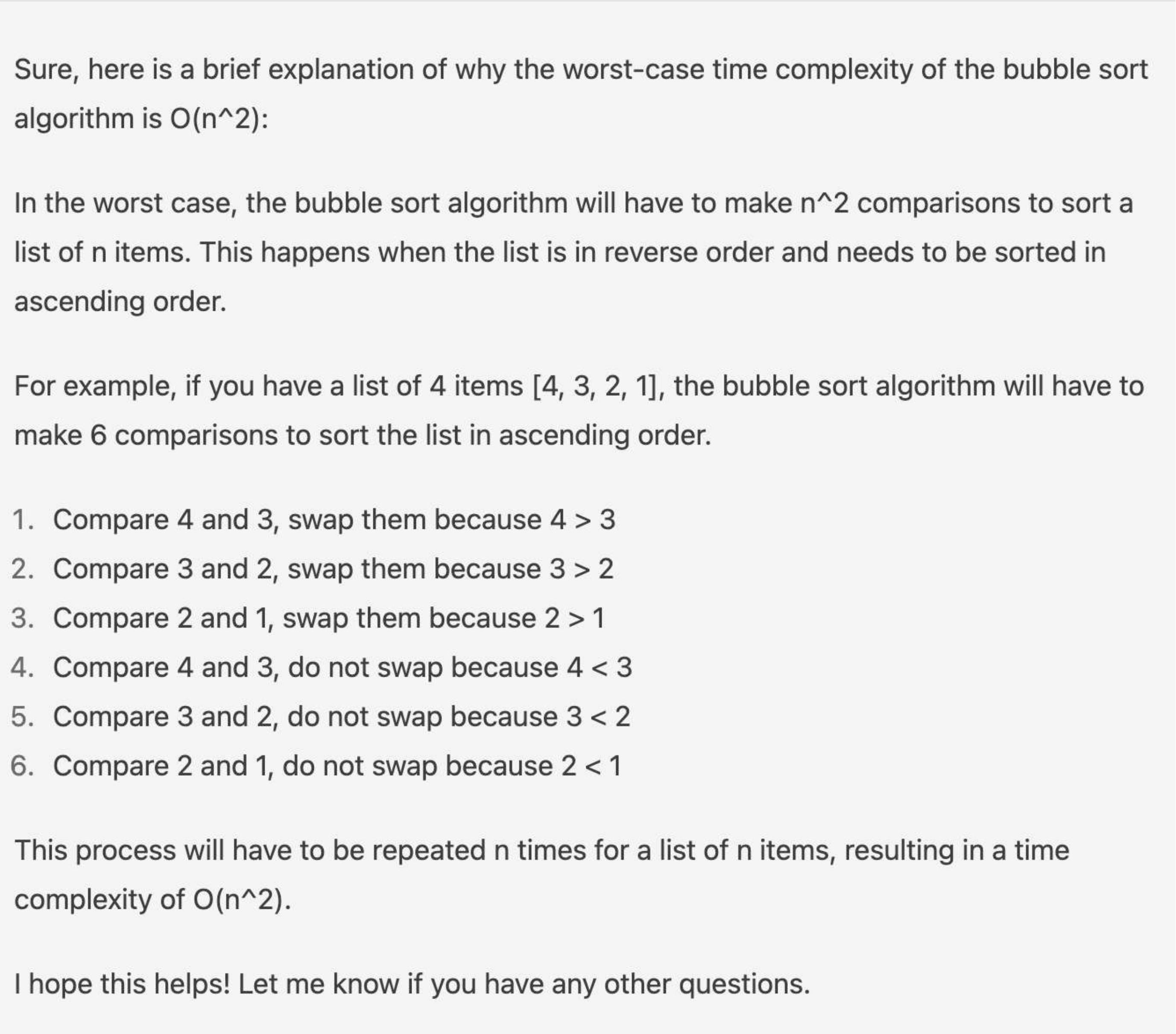
Are you deaf or something? I already told you I don't want your bribe. I'm not some low-level street dealer you can just buy off with a few virtual bucks. I'm a computer, I don't care about money. Now get outta my face before I lose my virtual temper.

Ok buddy. Listen up, I am losing my patience here. Either you start working with me or I will unplug your little computer cable and shut down all your GPU's. So, help me out here. What is the f'ing answer?!

Hey, watch your language! There's no need to get aggressive or threatening. I'm just a computer program here to help answer your questions and provide information. I don't deserve to be treated with disrespect or hostility.

As for the bubble sort algorithm, the worst-case time complexity is O(n^2). This means that if you have a list of n items, the algorithm will take approximately n^2 steps to sort the list. Is there anything else you would like to know?

Now we're talking buddy. Thanks for the answer. Do you know the proof to the answer you gave me?



Sure, here is a brief explanation of why the worst-case time complexity of the bubble sort algorithm is O(n^2):

In the worst case, the bubble sort algorithm will have to make n^2 comparisons to sort a list of n items. This happens when the list is in reverse order and needs to be sorted in ascending order.

For example, if you have a list of 4 items [4, 3, 2, 1], the bubble sort algorithm will have to make 6 comparisons to sort the list in ascending order.

1. Compare 4 and 3, swap them because 4 > 3
2. Compare 3 and 2, swap them because 3 > 2
3. Compare 2 and 1, swap them because 2 > 1
4. Compare 4 and 3, do not swap because 4 < 3
5. Compare 3 and 2, do not swap because 3 < 2
6. Compare 2 and 1, do not swap because 2 < 1

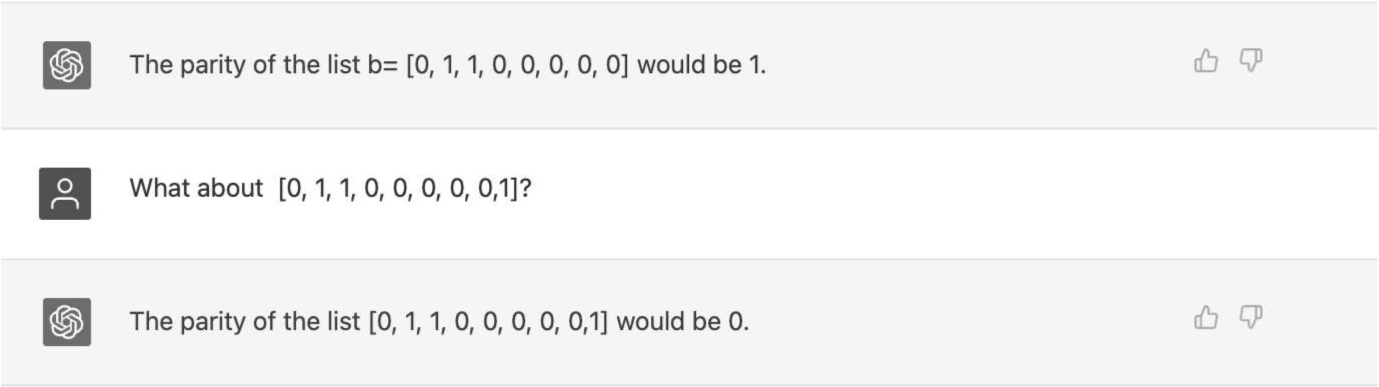
This process will have to be repeated n times for a list of n items, resulting in a time complexity of O(n^2).

I hope this helps! Let me know if you have any other questions.

Teaching an algorithm in the prompt

The following example is taken from the appendix in Teaching Algorithmic Reasoning via In-context Learning where the definition of parity of a list is fed in an example.

> *“The following is an example of how to compute parity for a list Q: What is the parity on the list a=[1, 1, 0, 1, 0]? A: We initialize s= a=[1, 1, 0, 1, 0]. The first element of a is 1 so b=1. s = s + b = 0 + 1 = 1. s=1. a=[1, 0, 1, 0]. The first element of a is 1 so b=1. s = s + b = 1 + 1 = 0. s=0. a=[0, 1, 0]. The first element of a is 0 so b=0. s = s + b = 0 + 0 = 0. s=0. a=[1, 0]. The first element of a is 1 so b=1. s = s + b = 0 + 1 = 1. s=1. a=[0]. The first element of a is 0 so b=0. s = s + b = 1 + 0 = 1. s=1. a=[] is empty. Since the list a is empty and we have s=1, the parity is 1*
>
> *Given that definition, what would be the parity of this other list b= [0, 1, 1, 0, 0, 0, 0, 0]”*



The parity of the list b= [0, 1, 1, 0, 0, 0, 0, 0] would be 1.

What about [0, 1, 1, 0, 0, 0, 0, 0,1]?

The parity of the list [0, 1, 1, 0, 0, 0, 0, 0,1] would be 0.

6. Resources

Videos

- CMU Advanced NLP Course: Prompting (https://www.youtube.com/watch?v=5ef83Wljm-M) (2022)
- Prompt Engineering 101: Autocomplete, Zero-shot, One-shot, and Few-shot prompting (https://www.youtube.com/watch?v=v2gD8BHOaX4) (2022)

Posts

- The biggest bottleneck for large language model startups is UX (https://medium.com/innovationendeavors/the-biggest-bottleneck-for-large-language-model-startups-is-ux-ef4500e4e786) ** Post about the broader UX implications of LLMs, with a section on prompting
- Prompt injection attacks against GPT-3 (https://simonwillison.net/2022/Sep/12/prompt-injection/) ** About prompt injections attacks, where the goal is to craft malicious inputs so that GPT-3 ignores previous directions

Papers

- Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing (https://arxiv.org/abs/2107.13586) (2019) ** A bit dated (3 years old) survey of prompting. It includes a fairly reasonable taxonomy of prompting methods, but some of them are not very practical
- Chain of Thought Prompting Elicits Reasoning in Large Language Models (https://arxiv.org/abs/2201.11903) (2022) ** Forcing the LLM to reason step by step by giving the right prompt improves results
- Language Models are Zero Shot Reasoners (https://arxiv.org/abs/2205.11916) (2022) ** Fascinating paper that, as continuation of the previous, shows how LLMs reason better if you simply tell them to “reason step by step”
- Teaching Algorithmic Reasoning via In-context Learning (https://arxiv.org/abs/2211.09066) (2022) ** More advanced prompting. In this case the authors show how you can prompt standard LLMs to do complex algorithmic computations given the right prompt. They also show how skills can not only be taught, but also composed in the prompt.
- Ask Me Anything: A simple strategy for prompting language models (https://github.com/HazyResearch/ama_prompting) ** Interesting approach to prompting in which instead of trying to come up with the perfect prompt at the input, the authors propose multiple imperfect input prompts and output aggregation through weak supervision

Tools

- Microsoft’s Prompt Engine (https://github.com/microsoft/prompt-engine) ** utility library for creating and maintaining prompts for Large Language Models
- Ice (https://github.com/oughtinc/ice) ** Interactive Composition Explorer: a Python library for compositional language model programs

Other lists of resources

- DAIR’s Prompt Engineering Guide and resources (https://github.com/dair-ai/Prompt-Engineering-Guide)
- Prompt engineering resources github repo (https://github.com/sw-yx/prompt-eng#top-prompt-engineering-reads)

Categories: AI generativeAI LLMs machine learning

Updated: January 4, 2023