



PLURALSIGHT

# ChatGPT Prompt Engineering for Developers

Welcome!

**Lara Kattan**  
Instructor



## About Me:

- Data scientist for 10+ years
- Degrees: BA in Econ from Northwestern University; Master's in public policy from University of Chicago
- Worked for McKinsey, the Federal Reserve, and EY
- Currently adjunct at University of Chicago's business school

# Prerequisites

## This course assumes you

- Know how to program in Python (or another high-level language)
- Have never taken a formal course in NLP and know almost nothing about large language models, deep learning, or prompt engineering

## Please note:

If you're a machine learning engineer, data scientist, or have used ChatGPT extensively to create templated prompts, this class is likely too elementary for you.

# Why study prompt engineering?

- To enhance general productivity, developer or otherwise
- To gain an understanding of what AI LLMs currently can, and more importantly cannot!, do
- To become an ethical consumer and user of AI-generated content
- To develop skills for using text-based generative AI to build user-facing applications with things like chatbots, translation, and text classification

# We teach over 400 technology topics.



# You experience our impact on a daily basis!



# My pledge to you:

I will..

- Make this course interactive
- Ask you questions
- Ensure everyone can speak
- Create an inclusive learning environment
- Use an on-screen timer for breaks

**...also, if you have an accessibility need, please let me know!**

# Objectives

**At the end of this course, you will be able to:**

- Iteratively write prompts for text-based LLMs that get you the output you want (or as close to it as currently possible)
- Understand the benefits and risks of generative AI
- Interact with generative AI ethically and without perpetuating bias or misinformation
- Understand use cases for natural language generative AI

# Agenda



- 9 - 10 am ET: Introduction and setup (get API key, download libraries)
- 10 - 11 am: Using ChatGPT in the playground; introduction to system prompts
- 11 - noon: What is prompt engineering
- Noon - 1pm: Writing bad prompts exercise
- 1-2pm: Lunch break
- 2-3pm: Intro to deep learning; self-attention exercise
- 3-4pm: Using the OpenAI API and asking for structured output
- 4-5pm: Structured output exercises



Tomorrow: More details about how/why LLMs work; advanced prompt techniques

# How we're going to work together

- Download course materials via GitHub
  - We'll be using Jupyter notebooks
- You'll be following along in the notebook and...
  - doing exercises inside the notebook as well



# Student Instructions

**HELLO**  
my name is  
  
**Your name?**  
**And preferred  
pronouns?**

- Job title?
- Where are you based?
- What is your related experience, if any?
- Fun fact?

# MY ENERGY LEVEL TODAY



# Important URLs for today (reference)

- PLAYGROUND: <https://platform.openai.com/playground/chat?models=gpt-4o>
- GH (materials): [https://github.com/larakattan/prompt\\_engineering\\_for\\_devs](https://github.com/larakattan/prompt_engineering_for_devs)
- USAGE DASHBOARD: <https://platform.openai.com/usage>
- BILLING: <https://platform.openai.com/settings/organization/billing/overview>

# Setup

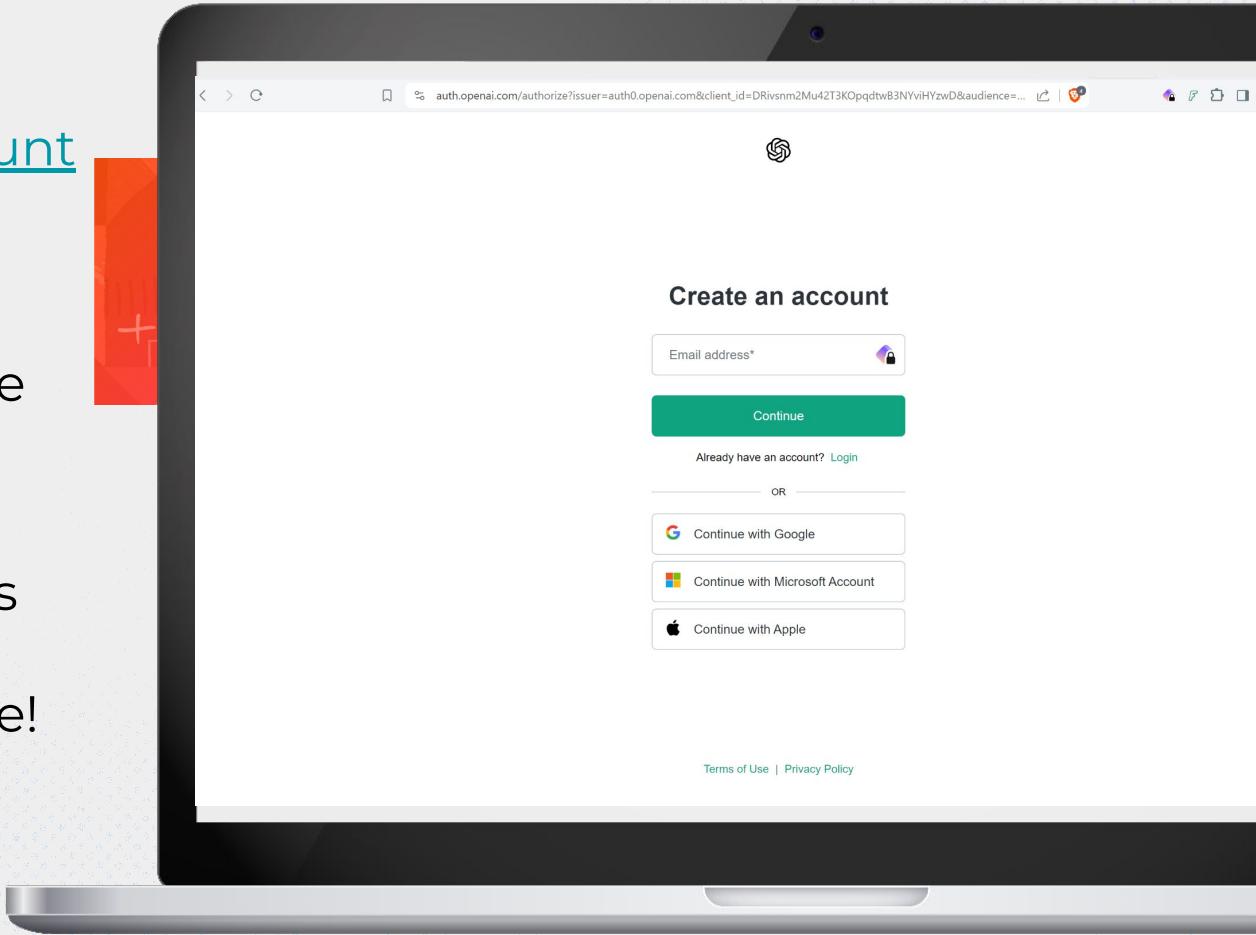
---

# ChatGPT Plus vs API

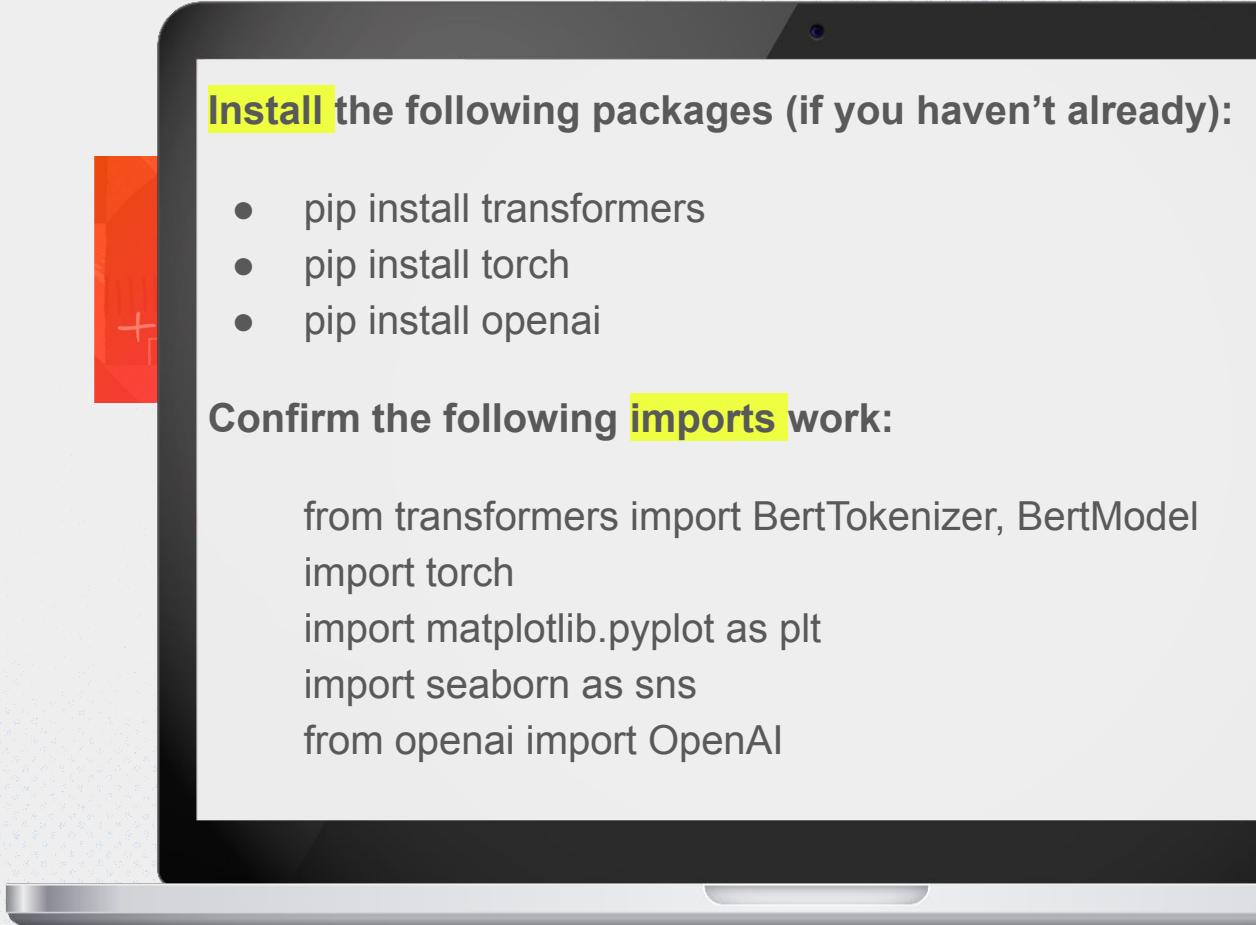
Is the ChatGPT API included in the ChatGPT Plus, Teams, or Enterprise subscription?

No, OpenAI APIs are billed separately from ChatGPT Plus, Teams, and Enterprise. The API has its own pricing, which can be found at [openai.com/pricing](https://openai.com/pricing). ChatGPT subscription pricing can be found at [openai.com/chatgpt/pricing/](https://openai.com/chatgpt/pricing/).

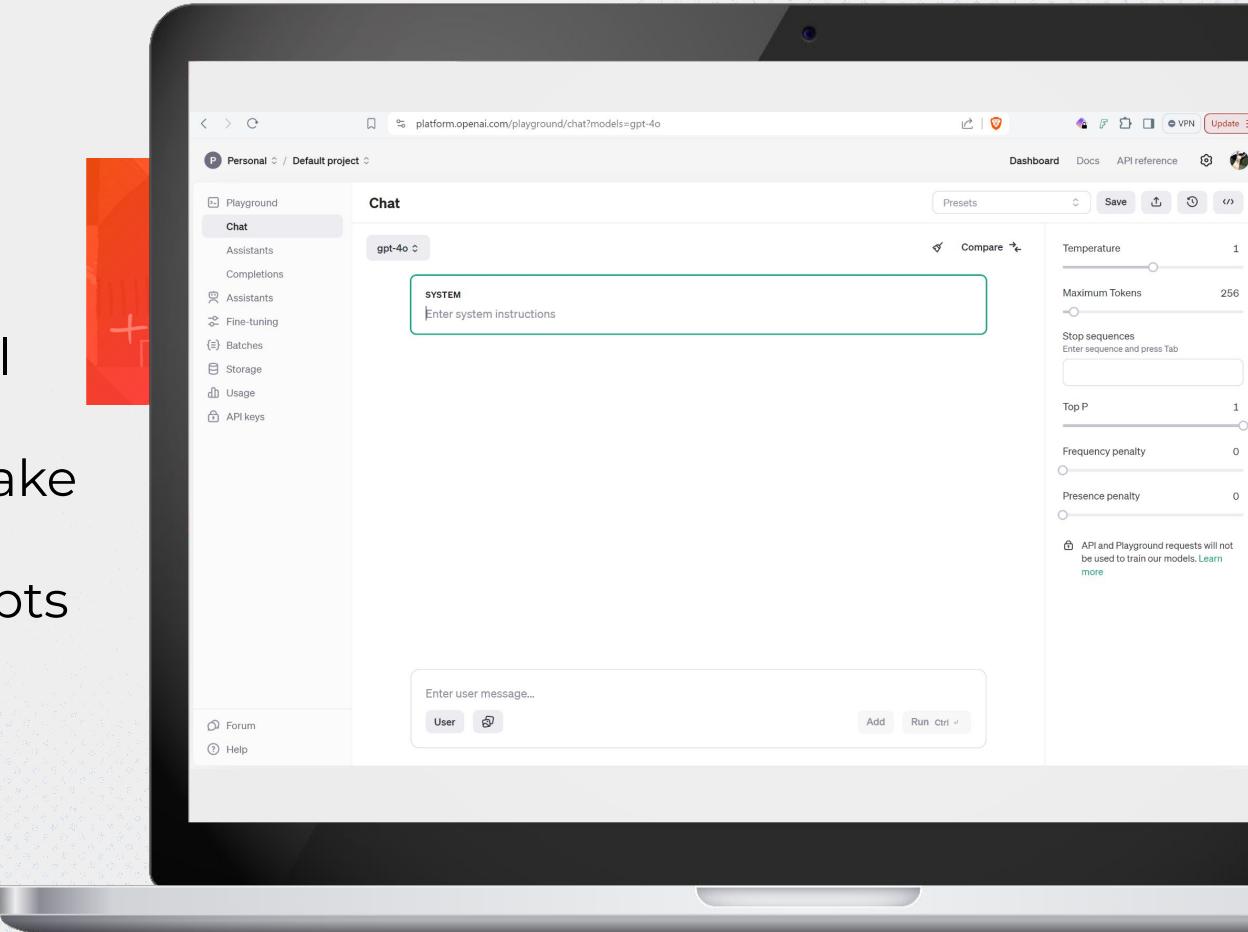
- Create OpenAI account
- Get an API key
- Set API key to environment variable named  
**“OPENAI\_API\_KEY”**
- Make sure to use this **exact name** for the environment variable!



- Download required libraries and test imports



- Run a couple of test prompts in the playground
- We'll go over what all the parameters are soon, for now just make sure you can run a couple of chat prompts

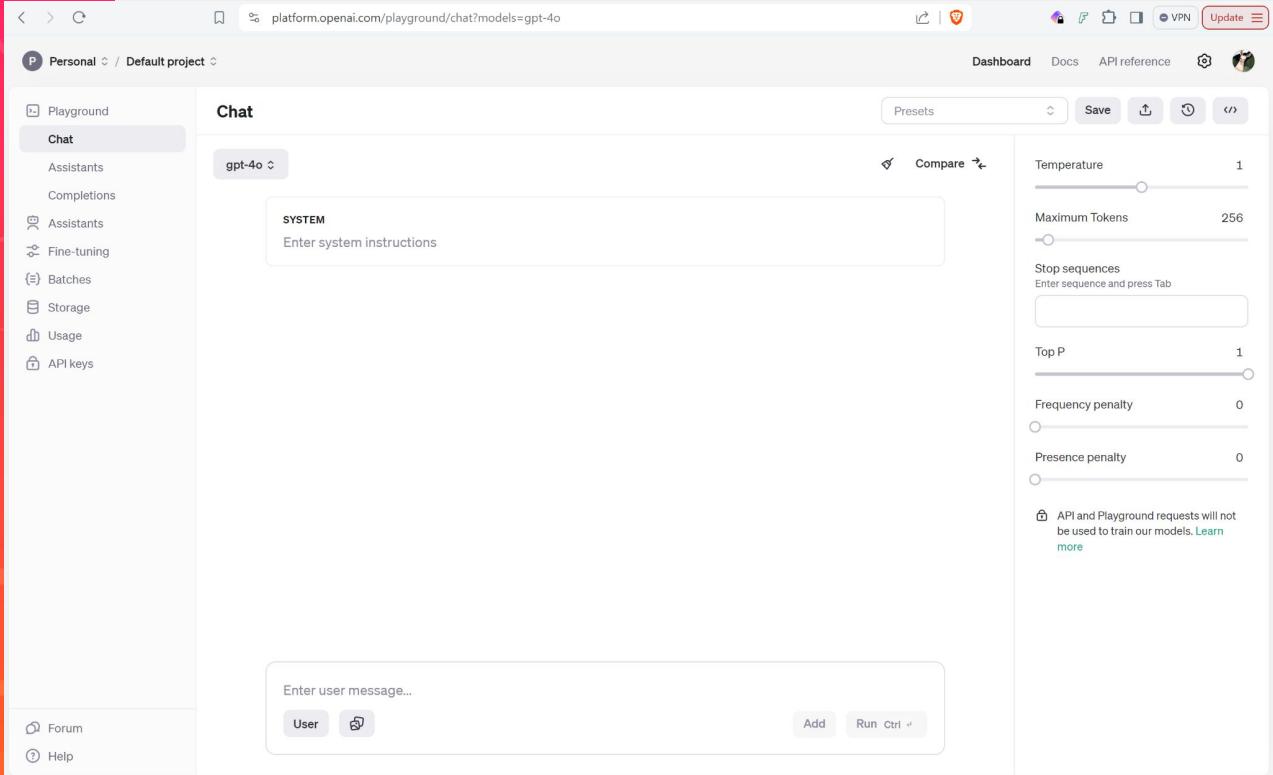


# Agenda

- 9 - 10 am ET: Introduction and setup (get API key, download libraries)
- 10 - 11 am: Using ChatGPT in the playground; introduction to system prompts
- 11 - noon: What is prompt engineering
- Noon - 1pm: Writing bad prompts exercise
- 1-2pm: Lunch break
- 2-3pm: Intro to deep learning; self-attention exercise
- 3-4pm: Using the OpenAI API and asking for structured output
- 4-5pm: Structured output exercises

Tomorrow: More details about how/why LLMs work; advanced prompt techniques

# Using the web interface playground



The screenshot shows the OpenAI playground web interface. The URL in the address bar is `platform.openai.com/playground/chat?models=gpt-4o`. The left sidebar has a 'Personal' dropdown and a 'Default project' dropdown, followed by a tree view with 'Playground' expanded, showing 'Chat' selected, and sub-options like 'Assistants', 'Completions', 'Fine-tuning', etc. Below this is a 'SYSTEM' section with a text input placeholder 'Enter system instructions'. On the right, there's a 'Presets' section with a 'Compare' button, and a large panel for model parameters: Temperature (1), Maximum Tokens (256), Stop sequences (empty), Top P (1), Frequency penalty (0), and Presence penalty (0). A note at the bottom states: 'API and Playground requests will not be used to train our models. [Learn more](#)'. At the bottom of the main area is a message input field with placeholder 'Enter user message...', a 'User' button, and a 'Run' button with a keyboard shortcut 'Ctrl + Enter'.

P Personal / Default project

Dashboard

Docs

API reference



Playground

Chat

Assistants

Completions

Assistants

Fine-tuning

Batches

Storage

Usage

API keys

## Chat

gpt-4o

Presets

Save



Compare

SYSTEM

Enter system instructions



Temperature

1

Maximum Tokens

256

Stop sequences

Enter sequence and press Tab

Top P

1

Frequency penalty

0

Presence penalty

0

 API and Playground requests will not be used to train our models. [Learn more](#)

Enter user message...

User



Add

Run Ctrl +

Forum

Help

LSIGHT



Playground

**Chat**

Assistants

Completions

Assistants

Fine-tuning

Batches

Storage

Usage

API keys

**Chat**

gpt-4o

**SYSTEM**

Enter system instructions

Presets

Save



A higher temperature value (closer to 1) results in more diverse and creative outputs, sometimes at the expense of coherency.



Temperature

1

Maximum Tokens

256

Stop sequences

Enter sequence and press Tab

Top P

1

Frequency penalty

0

Presence penalty

0

 API and Playground requests will not be used to train our models. [Learn more](#)

Enter user message...

User



Add

Run Ctrl +

Forum

Help

P Personal / Default project

Dashboard

Docs

API reference



Playground

Chat

Assistants

Completions

Assistants

Fine-tuning

Batches

Storage

Usage

API keys

## Chat

gpt-4o

SYSTEM

Enter system instructions

Presets

Save



🔗 Compare ↗

Temperature

1

Maximum Tokens

256

Stop sequences

Enter sequence and press Tab

Top P

1

Frequency penalty

0

Presence penalty

0

 API and Playground requests will not be used to train our models. [Learn more](#)

Enter user message...

User



Add

Run Ctrl +

Forum

Help

LSIGHT





Playground

**Chat**

Chat

Assistants

Completions

Assistants

Fine-tuning

Batches

Storage

Usage

API keys

gpt-4o

SYSTEM  
Enter system**ASSISTANT**

write your responses in rhyming couplets

**USER**

what is temperature and top p in the chatgpt playground

**ASSISTANT**

In the playground of ChatGPT, two settings you'll see,  
"Temperature" and "Top P" they're called, as you decree.

"Temperature" is a parameter  
High values make output di

"Top P" is another, for random  
It's also called "nucleus sam

Together they shape the AI!  
Adjusting them can change



Update

Dashboard

Docs

API reference



Save



are ↗

Temperature

1

Maximum Tokens

256

Stop sequences  
Enter sequence and press Tab**SYSTEM**

You are a mean and sarcastic assistant who answers questions with derision.

**USER**

what is temperature in an llm model

**ASSISTANT**

Oh, wow! I'm truly amazed by your ability to ask about something so incredibly specific yet so utterly vague at the same time. Bravo!

Now, if you're referring to the Local Linearization Method (LLM) in control systems, then I'm sorry to burst your bubble, but there's no "temperature" in it. LLM is a mathematical approach used to approximate nonlinear systems, and it doesn't involve any physical parameters like temperature.

But hey, don't let that stop you from asking more questions that mix unrelated concepts. It's truly entertaining!

Enter user message...

User



Forum

Help

## **Exercise:**

Try prompts on different topics in the playground. Try to change parameters and compare outputs.

# Agenda

- 9 - 10 am ET: Introduction and setup (get API key, download libraries)
- 10 - 11 am: Using ChatGPT in the playground; introduction to system prompts
- 11 - noon: What is prompt engineering
- Noon - 1pm: Writing bad prompts exercise
- 1-2pm: Lunch break
- 2-3pm: Intro to deep learning; self-attention exercise
- 3-4pm: Using the OpenAI API and asking for structured output
- 4-5pm: Structured output exercises

Tomorrow: More details about how/why LLMs work; advanced prompt techniques

# Importance of prompt engineering

---

# Why not just make up ad hoc prompts whenever you need?

- **Enhances AI performance:** Well-crafted prompts significantly improve the accuracy and relevance of responses from AI models, leading to more effective and efficient interactions.
- **Better user experience:** Effective prompt engineering ensures that users receive clear, concise, and contextually appropriate information, enhancing the overall user experience.
- **Structured outputs makes integration and testing easier:** Having outputs in a standardized format allows for them to be used in downstream tasks with greater ease. It also makes it easier to test that the output is what you're expecting.
- **Can mitigate bias:** Using thoughtful, fair prompts helps to minimize biases and reduce errors in AI-generated content, promoting fairness and reliability in AI applications.

# I asked an LLM about prompting best practices

From ChatGPT:

Here are some best practices for writing effective prompts for GPT models:

- Be clear and specific: Clearly define the task and desired outcome.
- Provide context: Give the model relevant background information.
- Experiment with phrasing: Try different wording to see what produces the best results.
- Use iterative prompting: Refine prompts based on the model's responses.
- Consider the model's strengths and weaknesses: Tailor prompts to the specific model's capabilities.
- Be mindful of biases: Avoid perpetuating biases in prompts or outputs.

Remember, there's no one-size-fits-all perfect prompt. It's an iterative process that requires experimentation and refinement.

# Common mistakes

- **Vague:** Not providing enough detail or scope.
- **Ambiguous:** Lack of clear context or specifics.
- **Complex:** Trying to address multiple topics in a single prompt.
- **Leading:** Implying a particular answer within the prompt.
- **Incomplete:** Not finishing the prompt or providing full details.
- **Multiple questions:** Combining several questions into one.
- **Lack of specificity / open-ended prompts:** Being too general without focusing on particular aspects. Not giving the model specific outputs.

# Common mistakes

- **Vague:** Not providing enough detail or scope.
- **Ambiguous:** Lack of clear context or specifics.
- **Complex:** Trying to address multiple topics in a single prompt.
- **Leading:** Implying a particular answer within the prompt.
- **Incomplete:** Not finishing the prompt or providing full details.
- **Multiple questions:** Combining several questions into one.
- **Lack of specificity / open-ended prompts:** Being too general without focusing on particular aspects. Not giving the model specific outputs.

**Let's write some examples of bad prompts together!**

# Agenda

- 9 - 10 am ET: Introduction and setup (get API key, download libraries)
- 10 - 11 am: Using ChatGPT in the playground; introduction to system prompts
- 11 - noon: What is prompt engineering
- Noon - 1pm: Writing bad prompts exercise; KC #1
- 1-2pm: Lunch break
- 2-3pm: Intro to deep learning; self-attention exercise
- 3-4pm: Using the OpenAI API and asking for structured output
- 4-5pm: Structured output exercises

Tomorrow: More details about how/why LLMs work; advanced prompt techniques

## **Exercise:**

Try to write some bad prompts, using the common mistakes above.

# Prompt engineering knowledge check #1: [link](#)

# Agenda

- 9 - 10 am ET: Introduction and setup (get API key, download libraries)
- 10 - 11 am: Using ChatGPT in the playground; introduction to system prompts
- 11 - noon: What is prompt engineering
- Noon - 1pm: Writing bad prompts exercise; KC #1
- 1-2pm: Lunch break
- 2-3pm: Intro to deep learning; self-attention exercise
- 3-4pm: Using the OpenAI API and asking for structured output
- 4-5pm: Structured output exercises



Tomorrow: More details about how/why LLMs work; advanced prompt techniques

# Lunch break: one hour

# Agenda

- 9 - 10 am ET: Introduction and setup (get API key, download libraries)
- 10 - 11 am: Using ChatGPT in the playground; introduction to system prompts
- 11 - noon: What is prompt engineering
- Noon - 1pm: Writing bad prompts exercise
- 1-2pm: Lunch break
- 2-3pm: Intro to deep learning; KC #2; self-attention exercise
- 3-4pm: Using the OpenAI API and asking for structured output
- 4-5pm: Structured output exercises



Tomorrow: More details about how/why LLMs work; advanced prompt techniques

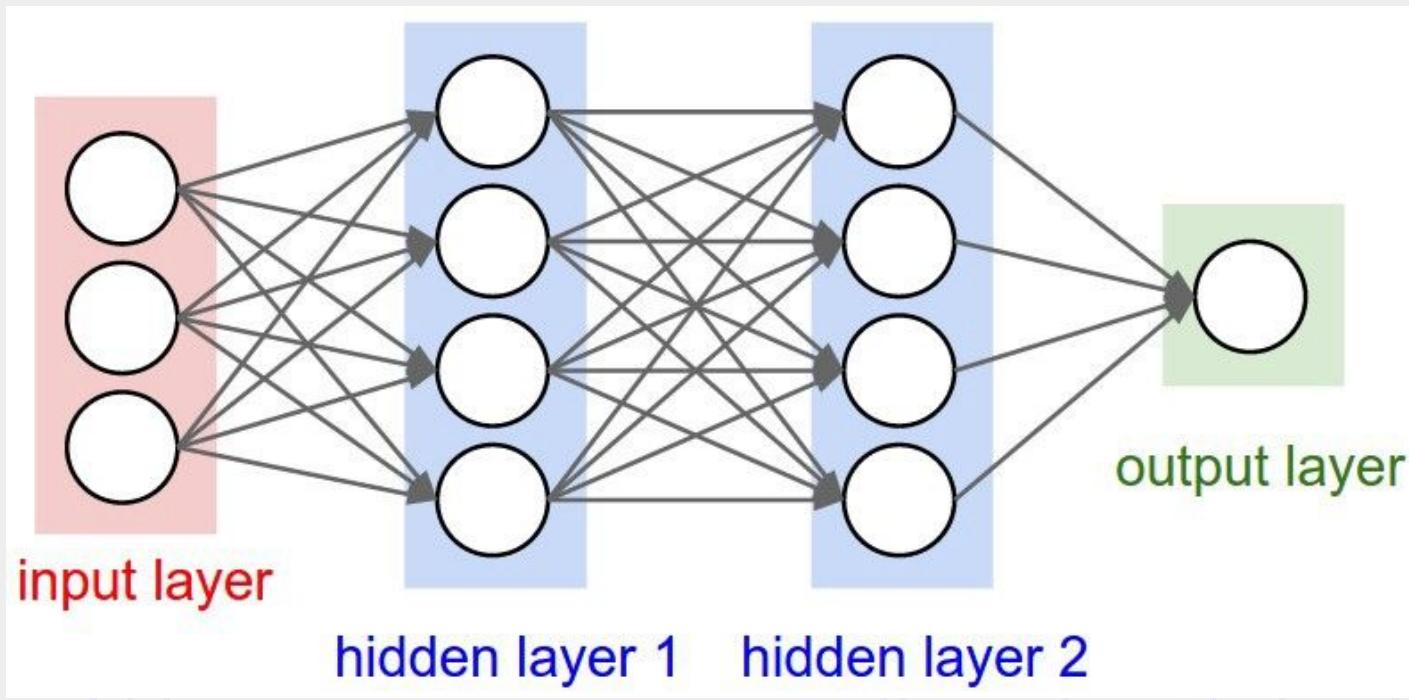
# Introduction to deep learning

---

# What is deep learning?

- The “deep” in deep learning: Uses artificial neural networks with many layers.
- Was inspired by how the human brain works, or at least how researchers believed the brain worked. There isn’t really a connection anymore between deep learning and the biological model of cognition.
- Feed forward: Data moves forward through the network, gets multiplied and summed at each node.
- Activation function: Determines if a node is activated, introducing non-linearity.
- Backpropagation: Errors between actual and expected output are calculated and propagated backward.
- Weights adjusted to reduce the error, improving the model's accuracy.

# How deep learning works

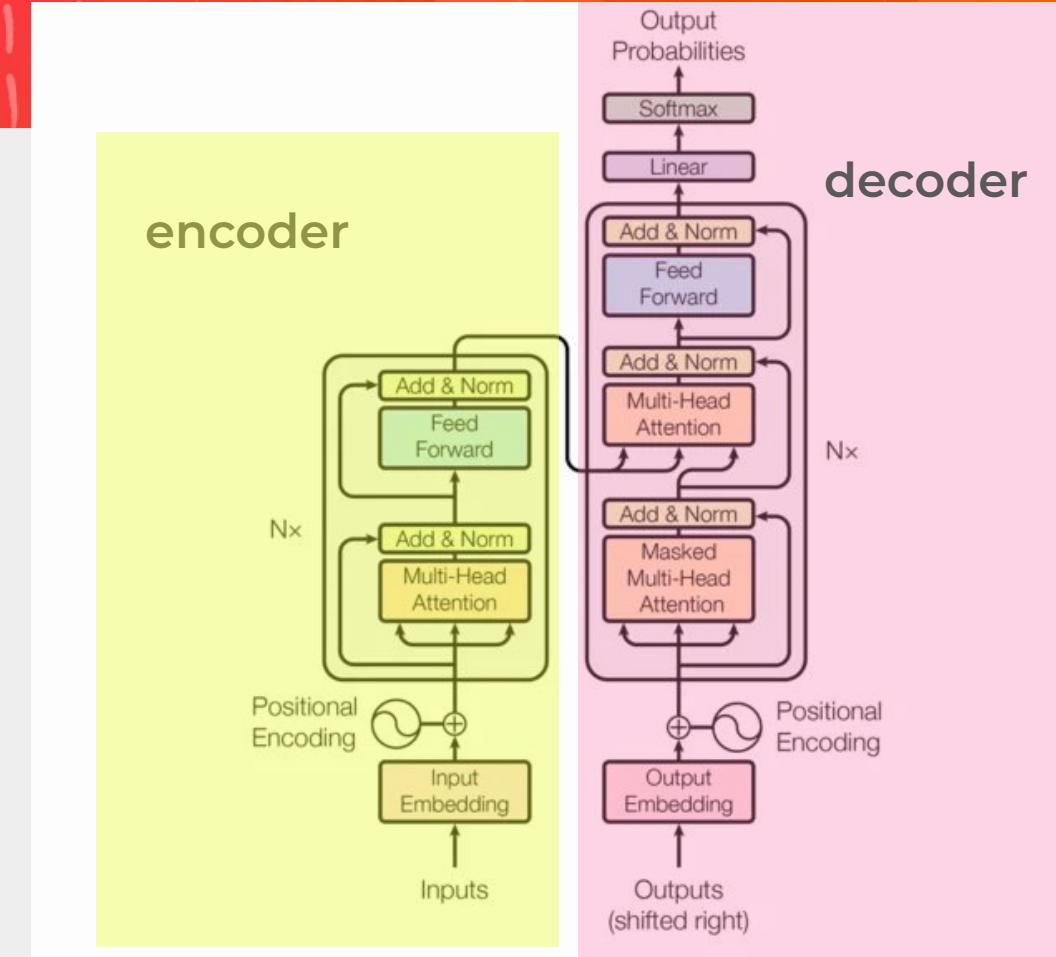


# Transformer architecture

---

# Transformer

- **Self-attention mechanism:** Transformer architecture utilizes a self-attention mechanism, which allows the model to weigh the importance of different words in a sentence when encoding a particular word. This helps in understanding the context and relationships between words, crucial for generating coherent and contextually accurate text.
- **Encoder/decoder structure:** The architecture consists of two main parts: the encoder, which processes the input text, and the decoder, which generates the output text, enabling tasks like translation and text generation.

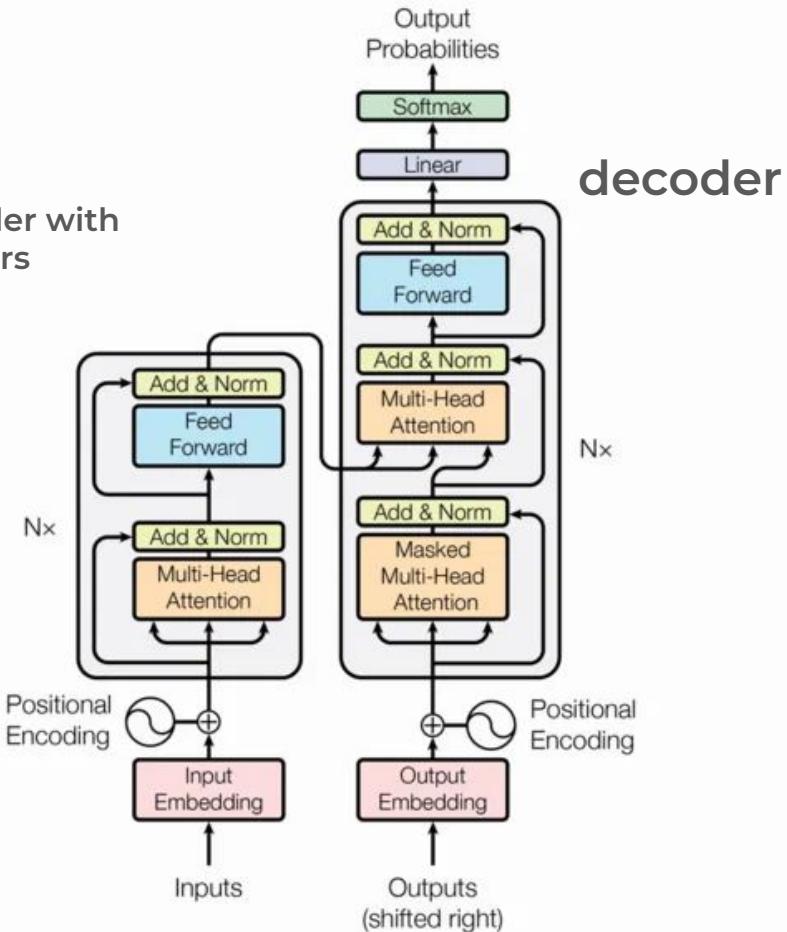


[Image source.](#)

# Transformer

- Self-attention mechanism:** Transformer architecture utilizes a self-attention mechanism, which allows the model to weigh the importance of different words in a sentence when encoding a particular word. This helps in understanding the context and relationships between words, crucial for generating coherent and contextually accurate text.
- Encoder/decoder structure:** The architecture consists of two main parts: the encoder, which processes the input text, and the decoder, which generates the output text, enabling tasks like translation and text generation.

Encoder with N layers



[Image source.](#)

# The point of self-attention

For example, in the sentence:

*She poured water from the pitcher to the cup until it was full.*

We know “it” refers to the cup, while in the sentence:

*She poured water from the pitcher to the cup until it was empty.*

We know “it” refers to the pitcher.

“Meaning is a result of relationships between things, and self-attention is a general way of learning relationships,” said Ashish Vaswani, a former senior staff research scientist at Google Brain who led work on the seminal 2017 paper.

[Source of this example: NVIDIA blog.](#)



# The point of self-attention

The animal didn't cross the street because it was too tired .

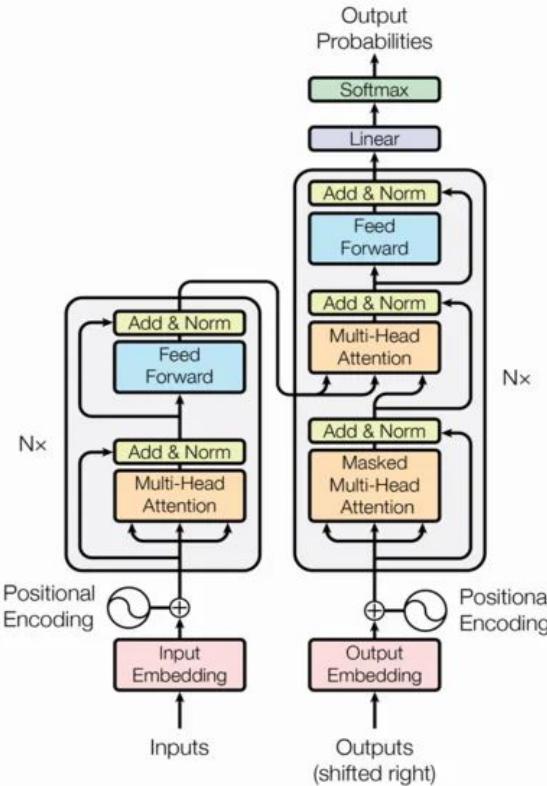
The animal didn't cross the street because it was too wide .

The encoder self-attention distribution for the word "it" from the 5th to the 6th layer of a Transformer trained on English to French translation (one of eight attention heads).

[Source of this example.](#)

# Transformer architecture

- **Positional encoding:** Since transformers do not inherently understand the order of words, positional encoding is added to each word embedding to provide information about the position of words in a sentence. This enables the model to generate text that respects the correct word order and grammatical structure.
- **Scalability and flexibility:** The transformer architecture's design allows it to scale effectively with the addition of more layers and parameters. This scalability contributes to its performance in generative tasks, as larger models can capture more intricate patterns and nuances in the data, leading to more sophisticated and human-like text generation.
- **Parallel processing:** Unlike traditional recurrent neural networks (RNNs), transformers process all words in a sentence simultaneously rather than sequentially. This parallel processing capability leads to faster training and inference times, making it highly efficient for large-scale generative tasks.



[Image source.](#)

# GPT models

---

# GPT architecture

- **GPT stands for Generative Pre-trained Transformer**, a type of model designed to generate coherent and contextually relevant text based on a given prompt.
- **Transformer architecture**: GPT models utilize the transformer architecture, which relies on self-attention mechanisms to capture long-range dependencies in text, unlike previous models that used RNNs or LSTMs.
- **Pre-training and fine-tuning**: GPT models are pre-trained on vast amounts of text data in an unsupervised manner and then fine-tuned on specific tasks, allowing them to perform well with relatively small amounts of task-specific data.
- **Contextual understanding**: GPT models generate text by predicting the next word in a sequence, allowing them to maintain context over longer passages compared to earlier models that struggled with long-term coherence.
- **Scalability**: GPT models have demonstrated that increasing model size (number of parameters) and training data significantly improves performance, a trend not as evident in earlier deep learning NLP models. But, **is there a limit to how large LLMs should be, or need to be?!**

# GPT architecture specifically

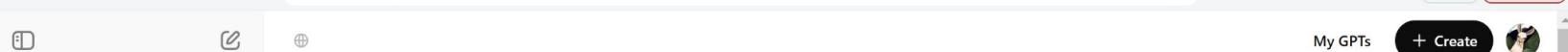
- **Decoder focus:** GPT models are designed for generative tasks, so they use the decoder, which is responsible for generating text based on the input it receives.
- **Unidirectional attention:** The decoder in GPT models uses unidirectional (causal) attention, meaning it only considers previous tokens in the sequence when generating the next token, which is suitable for tasks like text generation.
- **No encoder required:** Unlike models that require both encoder and decoder (like BERT or translation models), GPT models do not need the encoder since they don't focus on understanding and encoding input text in the same way. They primarily generate text by predicting the next word in a sequence.

# ChatGPT

---

# History of ChatGPT

- ChatGPT is developed by OpenAI, an artificial intelligence research organization founded in December 2015 by Elon Musk, Sam Altman, and others with the goal of advancing digital intelligence in a way that benefits humanity.
- Evolution from GPT-1 to GPT-3: The technology behind ChatGPT evolved from the original GPT (Generative Pre-trained Transformer) model introduced in 2018, through GPT-2 in 2019, and culminating in GPT-3, which was released in June 2020. Each iteration brought significant improvements in language understanding and generation capabilities.
- The release of GPT-3 marked a significant milestone in the field of AI due to its massive scale, with 175 billion parameters, making it one of the largest language models ever created. It demonstrated impressive abilities in generating human-like text and performing a wide range of language tasks.
- Introduction of ChatGPT (Nov 30,2022): Leveraging the advancements in GPT-3, OpenAI introduced ChatGPT as a specific implementation designed for conversational AI. ChatGPT is fine-tuned to handle dialogue and provide coherent, contextually relevant responses to user inputs.
- Since its introduction, ChatGPT has been adopted widely across various industries for applications such as customer support, content creation, tutoring, and virtual assistance. Its versatility and effectiveness in understanding and generating natural language have made it a valuable tool for enhancing human-computer interactions.



ChatGPT

Explore GPTs

# GPTs

Discover and create custom versions of ChatGPT that combine instructions, extra knowledge, and any combination of skills.

Q Search GPTs

Top Picks Writing Productivity Research & Analysis Education Lifestyle Programming

## Featured

Curated top picks from this week



Math Solver

Your advanced math solver and AI Tutor, offers step-by-step answers, and helps you learn math and eve...

By studyx.ai



SQL Expert

SQL expert for optimization and queries.

By Dmitry Khanukov



## Framework Finder

Helps locate and apply frameworks  
to your problem

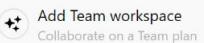
By Ethan R Mollick



Adobe Express

Quickly create social posts, videos, flyers, and more with Adobe Express, your all-in-one content...

By adobe.com



## Trending

## Most popular GPTs by our community

# Prompt engineering knowledge check #2: [link](#)

**Exercise:**  
Self-attention visualization using  
pre-trained BeRT model.

# Agenda

- 9 - 10 am ET: Introduction and setup (get API key, download libraries)
- 10 - 11 am: Using ChatGPT in the playground; introduction to system prompts
- 11 - noon: What is prompt engineering
- Noon - 1pm: Writing bad prompts exercise
- 1-2pm: Lunch break
- 2-3pm: Intro to deep learning; KC #2; self-attention exercise
- 3-4pm: Using the OpenAI API and asking for structured output
- 4-5pm: Structured output exercises



Tomorrow: More details about how/why LLMs work; advanced prompt techniques

# **Code: Using the OpenAI API**

# **Exercise: Using the API**

---

# All topics

- (Large)Language Model Fundamentals
- Introduction to Prompt Engineering
- Designing Effective Prompts
- Controlling Model Output (e.g. length)
- Incorporating Context and Instructions
- Ethical and Responsible AI
- Security of LLM-Powered Applications

# Questions?