

香港城市大學
City University of Hong Kong

Lecture 5: (Team-based Learning)

Advanced Topics and Recent Trends

Large Language Models for Data Engineering

CS5481 Data Engineering

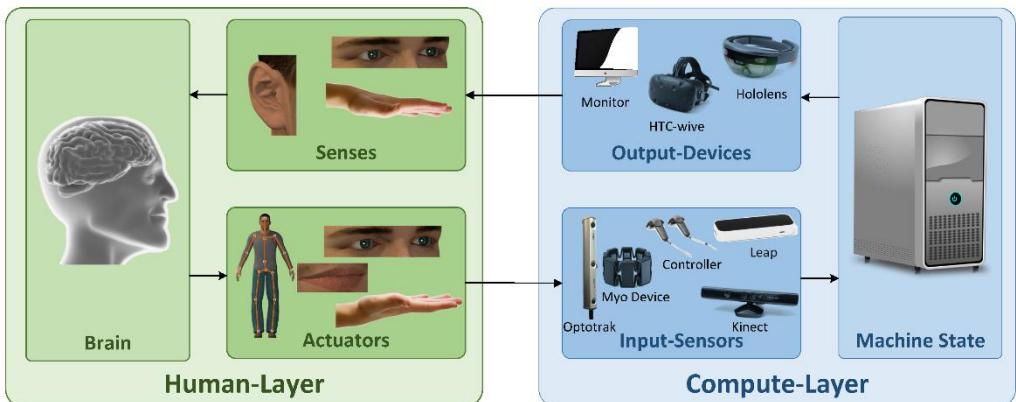
Instructor: Linqi Song

Outline

1. LLM&Prompt Engineering
2. Large Language Models for Data Preprocessing
3. Large Language Models for Data Annotation
4. Large Language Models for Data Augmentation
5. Large Language Models for Data Visualization
6. Large Language Models for Text Analysis

ChatGPT becomes the the fastest-growing 'app' of all time

- ChatGPT, Chat Generative Pre-trained Transformer, is an artificial intelligence (AI) chatbot developed by OpenAI and released in November 2022.
- ChatGPT can communicate with human beings in a more **natural way**.



Examples	Capabilities	Limitations
"Explain quantum computing in simple terms" →	Remembers what user said earlier in the conversation	May occasionally generate incorrect information
"Got any creative ideas for a 10 year old's birthday?" →	Allows user to provide follow-up corrections	May occasionally produce harmful instructions or biased content
"How do I make an HTTP request in Javascript?" →	Trained to decline inappropriate requests	Limited knowledge of world and events after 2021

Send a message... 

ChatGPT Mar 23 Version. Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts.

ChatGPT has powerful textual processing abilities

- ChatGPT is versatile. It can **write and debug computer programs, mimic the style of celebrity CEOs and write business pitches, compose music, teleplays, fairy tales and student essays, answer test questions** (sometimes, depending on the test, at a level above the average human test-taker), **write poetry and song lyrics, emulate a Linux system; simulate entire chat rooms, play games like tic-tac-toe and simulate an ATM.**

so please write python codes to scrape news from financial times

Sure, here's an example Python code that uses BeautifulSoup and Requests to scrape news from the Financial Times:

```
python
import requests
from bs4 import BeautifulSoup

# Set the URL of the Financial Times news page you want to scrape
url = 'https://www.ft.com/'

# Send a request to the URL and get the HTML response
response = requests.get(url)
html = response.content

# Parse the HTML using BeautifulSoup
soup = BeautifulSoup(html, 'html.parser')
```

so introduce yourself

Hello! I am ChatGPT, a language model trained by OpenAI, based on the GPT-3.5 architecture. I am designed to generate human-like responses to natural language inputs, and I am constantly learning and improving my ability to understand and interact with people. My goal is to provide helpful and informative responses to any questions or requests you have.

so I want to give a talk to management science professors about how to use ChatGPT to assist their research and teaching. Can you help to provide some topics to cover?

so Certainly! Here are some topics you may want to cover in your talk:

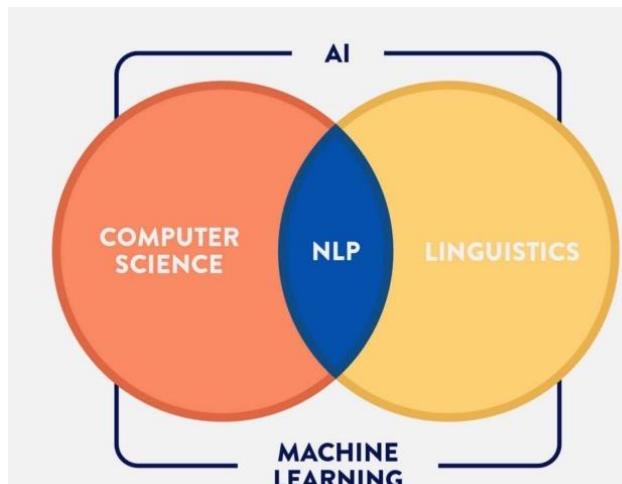
1. Introduction to ChatGPT: Start by introducing ChatGPT and explaining what it is, how it works, and what its capabilities are. This will give your audience a basic understanding of the technology and set the stage for the rest of your talk.
2. Research Applications: Discuss the various ways in which ChatGPT can be used in management science research, such as natural language processing, sentiment analysis, and data analysis. Explain how ChatGPT can help researchers to gain new insights into complex problems and make more accurate predictions.
3. Teaching Applications: Explain how ChatGPT can be used as a teaching tool in management science classrooms, such as for generating case studies, creating quizzes, or even assisting in grading. You may want to provide examples of how ChatGPT can be integrated into existing teaching methods.
4. Data Collection: Discuss how ChatGPT can be used to collect data from various sources, such as social media or customer reviews. Explain how this data can be used to gain insights into customer behavior, market trends, and more.
5. Limitations and Ethical Concerns: Be sure to mention the limitations of ChatGPT, such as its tendency to generate biased or inaccurate responses. Discuss the ethical concerns that arise when using ChatGPT, such as the potential for misuse or abuse of the technology.
6. Future Developments: Finally, discuss the potential for future developments in ChatGPT and how they might impact management science research and teaching. You may want to discuss new features, improvements in accuracy, or the development of new applications.

By covering these topics, you will provide a comprehensive overview of the ways in which ChatGPT can be used in management science research and teaching, while also highlighting its limitations and ethical concerns. Good luck with your talk!

Regenerate response

Natural Language Processing

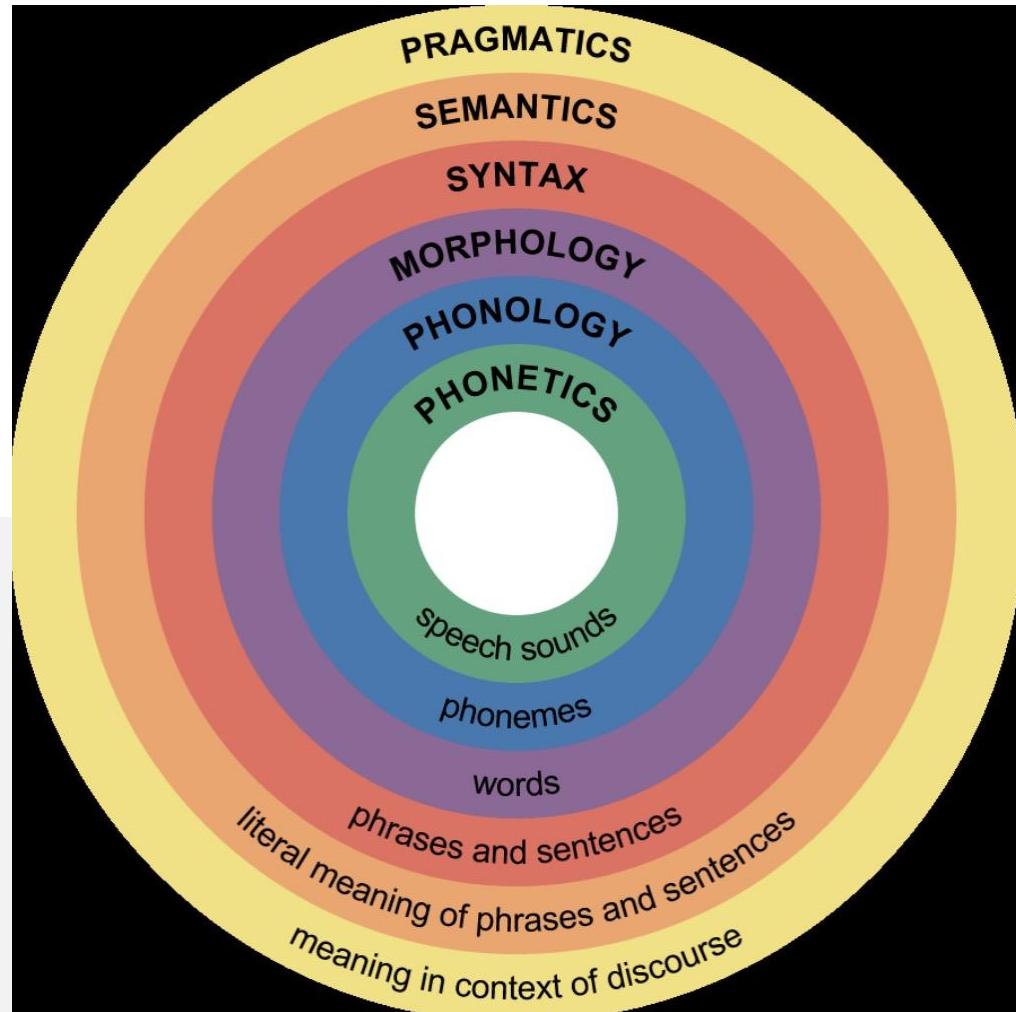
- The core technique behind ChatGPT is Natural Language Processing (NLP). NLP is an important branch of AI. It is being considered the 'Jewel of the AI crown'.
- Make computers to learn, process and manipulate natural languages to interact with humans.



The Jewel of the Artificial Intelligence Crown

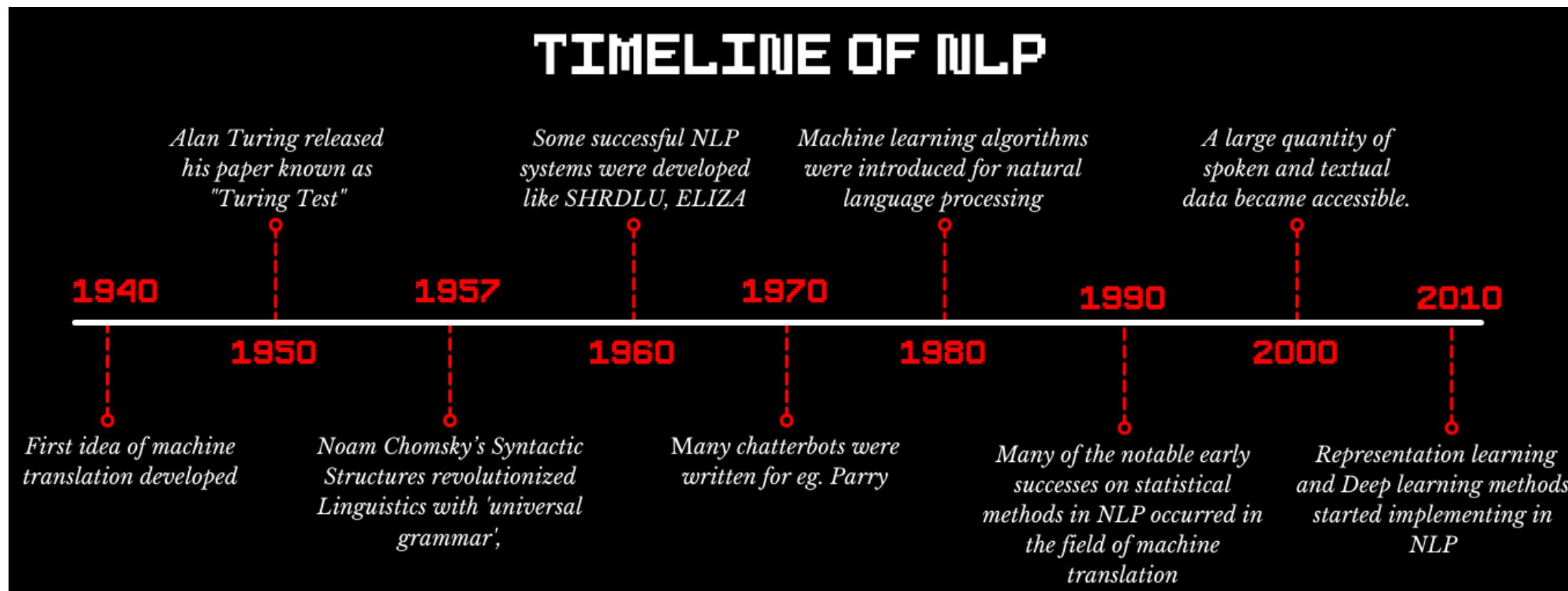


aNumak & Company
Jan 13, 2022 · 5 min read · [Listen](#)



A short history of NLP

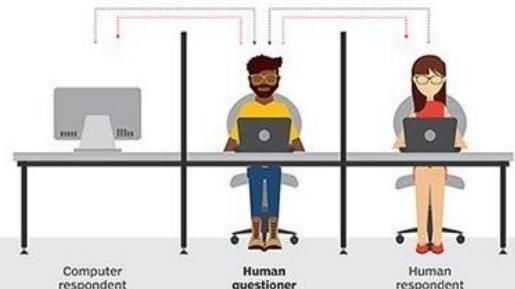
TIMELINE OF NLP



Turing test

During the Turing test, the human questioner asks a series of questions to both respondents. After the specified time, the questioner tries to decide which terminal is operated by the human respondent and which terminal is operated by the computer.

QUESTION TO RESPONDENTS ANSWERS TO QUESTIONER

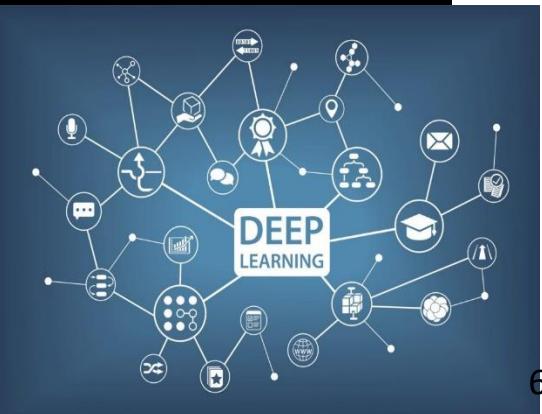
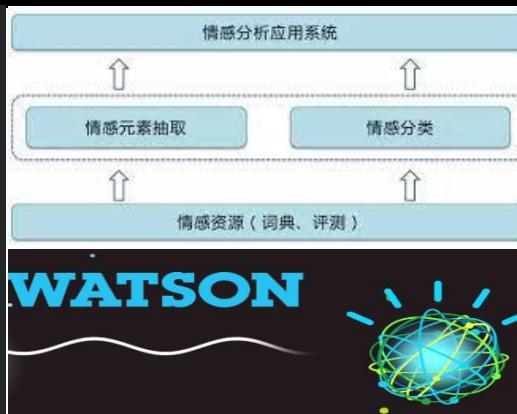


Welcome to

EEEEE: LL IIII ZZZZZ: AAAAA
EE: LL II ZZ AA AA
EEERE: LL II ZZZ: AAAAAAA
EE: LL II ZZ AA AA
EEEEE: LLLLLL IIII ZZZZZ: AA AA

Eliza is a mock Rogerian psychotherapist. The original program was described by Joseph Weizenbaum in 1966. This implementation by Norbert Landsteiner 2005.

ELIZA: Is something troubling you ?
YOU: Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU: They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU: Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here ?
YOU: He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU: It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy ?
YOU:



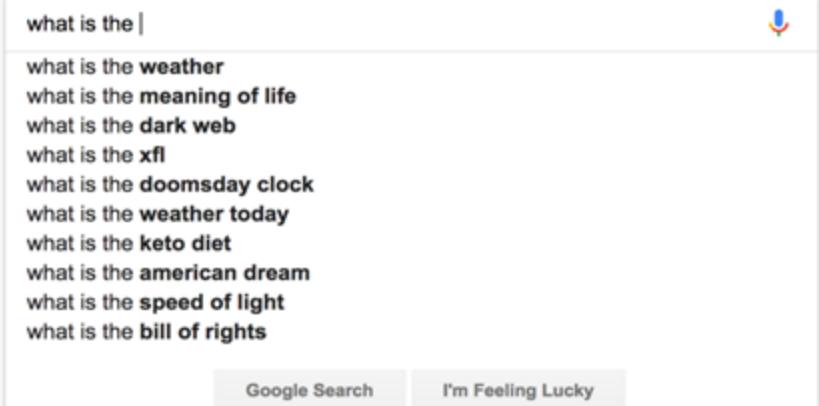
What is a language model?

- A language model is a probability distribution over sequences of words. Given such a sequence, say of length T , it assigns a probability $P(x^{(1)}, x^{(2)}, \dots x^{(T)})$ to the whole sequence.
- To determine whether a given ordering of words sounds like natural language
- Given such a sequence of length T , it assigns a probability $P(x^{(1)}, x^{(2)}, \dots x^{(T)})$ to the whole sequence. For example, consider a piece of text $(x^{(1)}, x^{(2)}, \dots x^{(T)})$, then according to the Language Model, the probability of the text could be written as:

$$\begin{aligned} P(x^{(1)}, \dots, x^{(T)}) &= P(x^{(1)}) \times P(x^{(2)} | x^{(1)}) \times \dots \times P(x^{(T)} | x^{(T-1)}, \dots, x^{(1)}) \\ &= \prod_{t=1}^T P(x^{(t)} | x^{(t-1)}, \dots, x^{(1)}) \end{aligned}$$

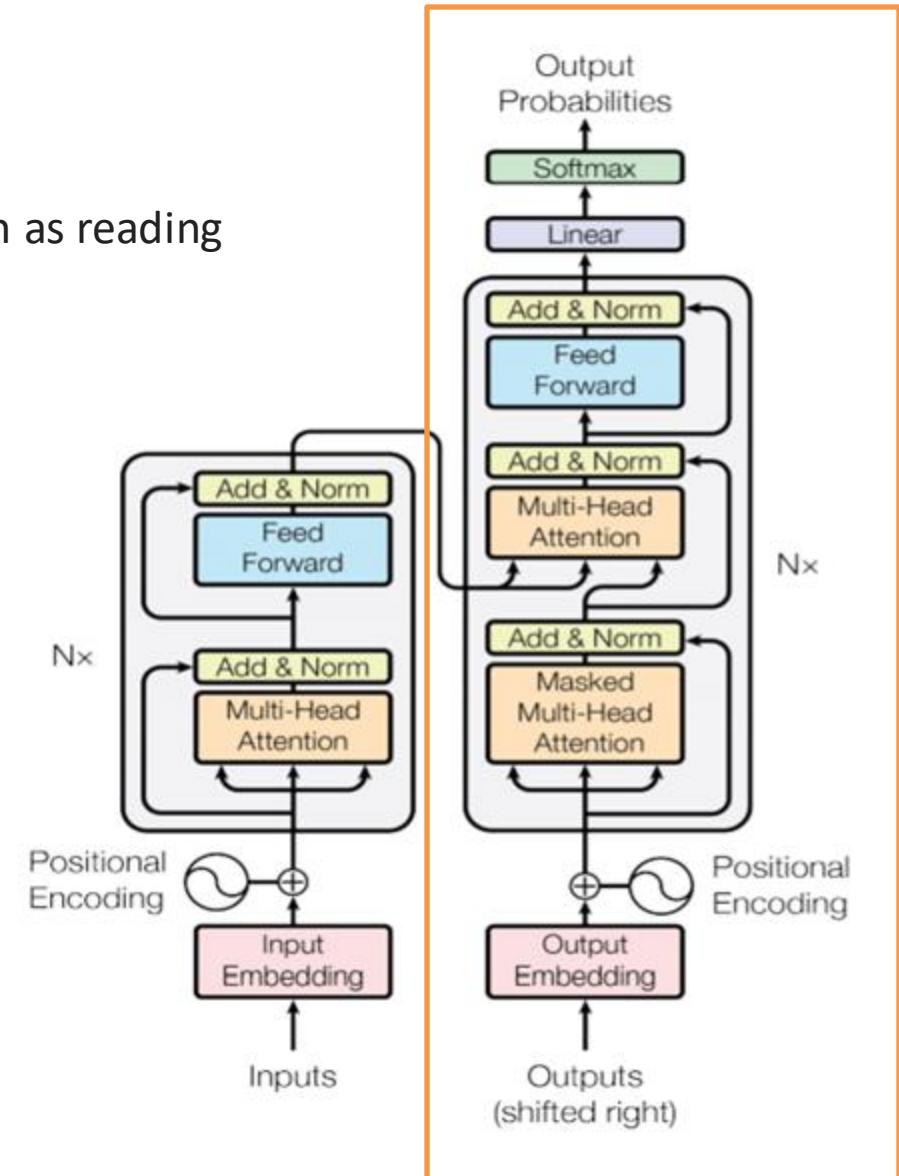
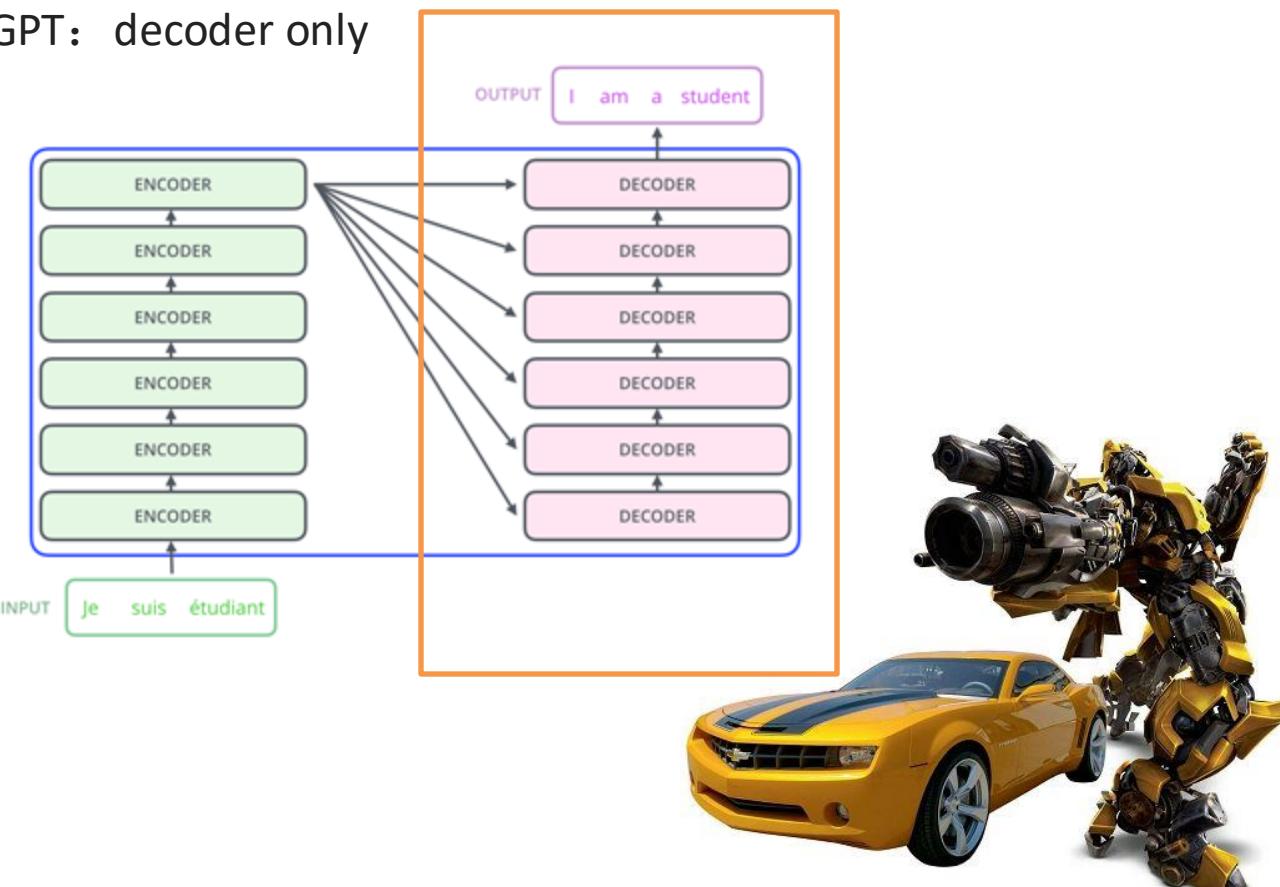


A language model can also be interpreted as predicting what word comes next



ChatGPT's core technique: Transformer

- Proposed by Google: Attention is All You Need (Vaswani et al., 2017)
- Main technique: **multi-head self attention mechanism**.
- Solve sequence to sequence understanding and generation tasks, such as reading comprehension, machine translation, text generations.
- GPT: decoder only



The evolution of GPT models



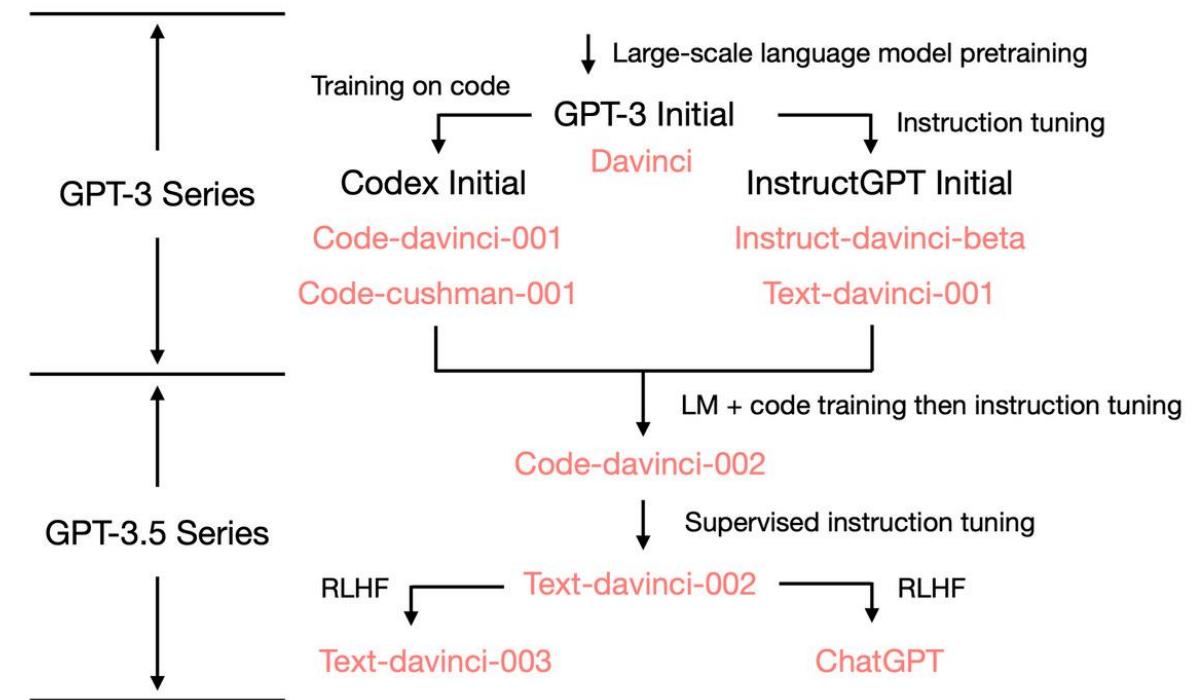
The first version of GPT was released

The second version of GPT was released

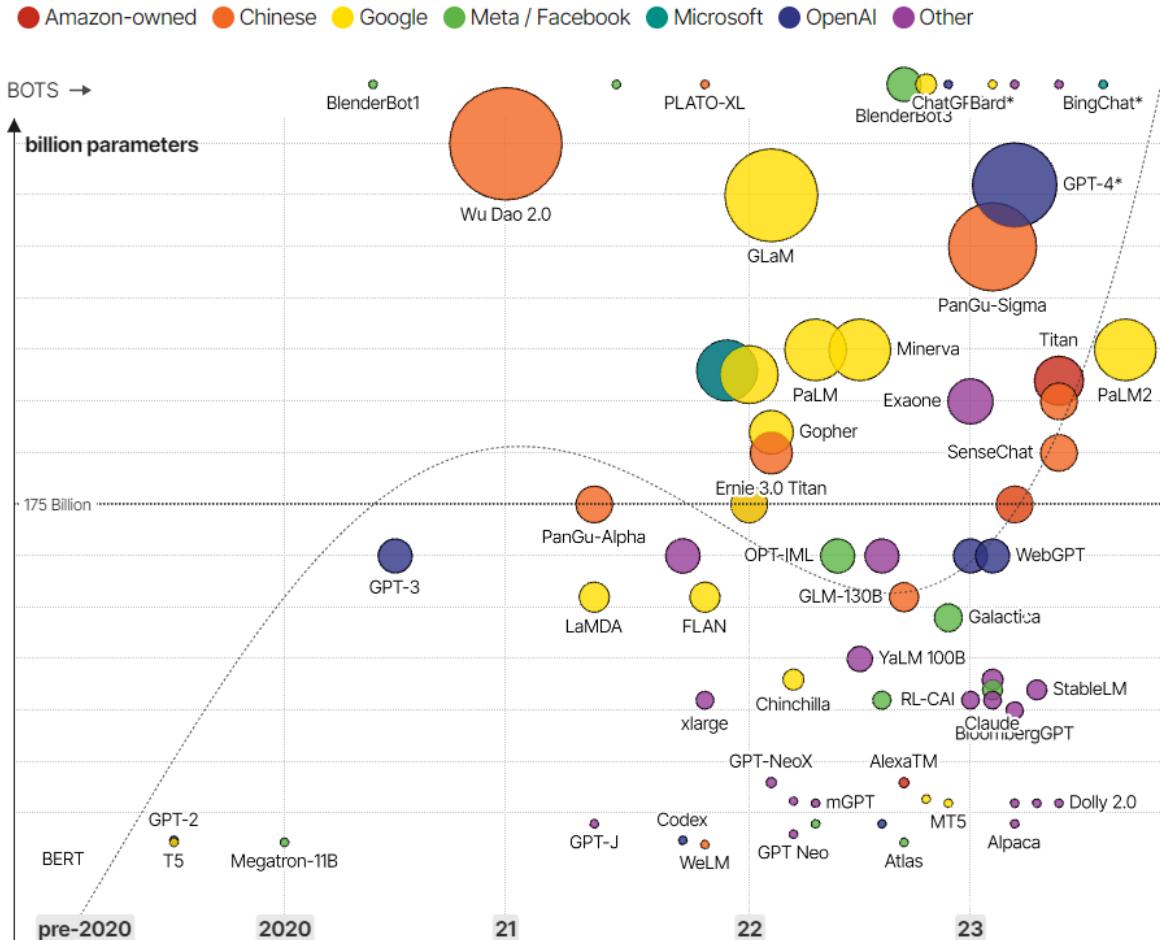
Initial GPT-3 preprint paper was published at arXiv. API became publicly available on Nov. 18th, 2021

ChatGPT was announced on OpenAI blog. ChatGPT API became available on Mar. 1st, 2023

GPT-4 was released via ChatGPT. API will be publicly available soon.



Background: The war of large language models!

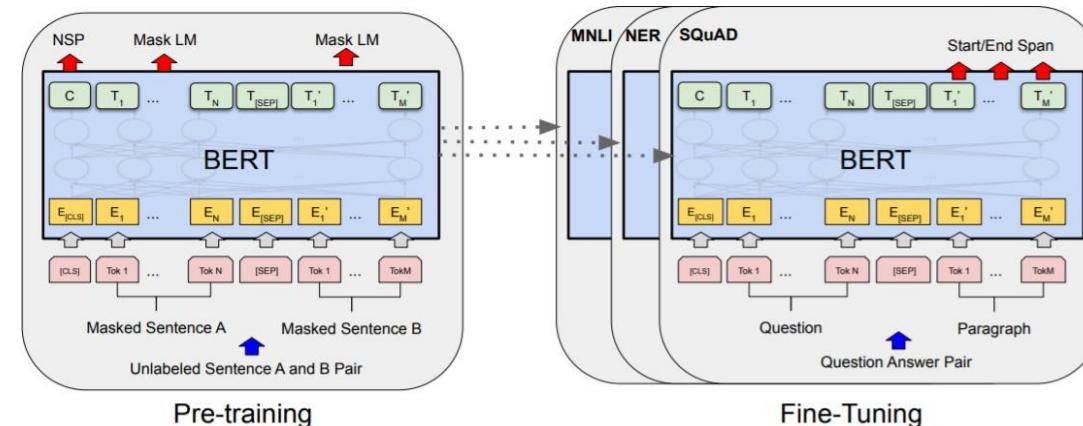


LLM ecosystem has been built up.
Tech giant: OpenAI, Google, Microsoft, ...
Open-source community: HuggingFace, Eleuther AI, Langchain, ...
Domain players: BloombergGPT, MedGPT, ...
End-customers and more

Pretraining-finetuning

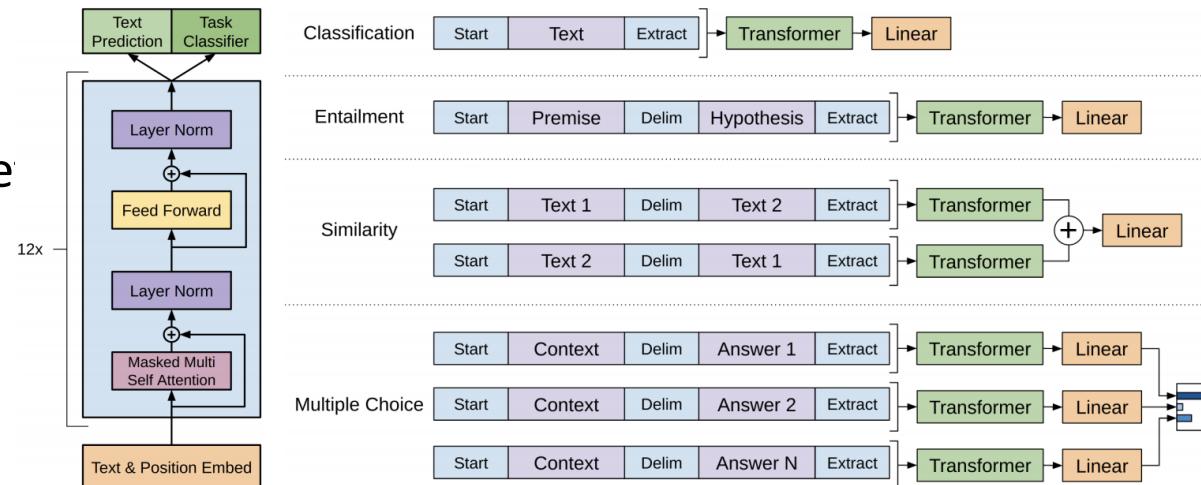
Google BERT ([Devlin et al., 2018](#); released in 2018/10)

- Encoder only structure
- Same model for training and downstream tasks
- Fine-tuning for downstream tasks



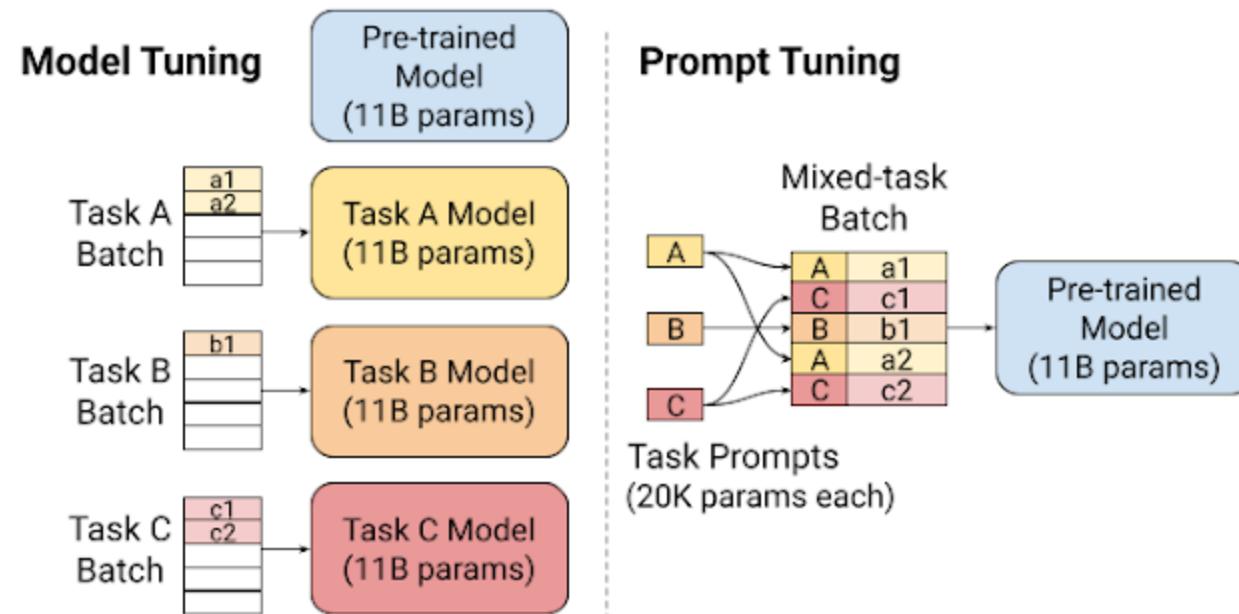
GPT (117M parameters; [Radford et al., 2018](#))

- Decoder only structure
- Same model for training and downstream tasks
- Language modeling at scale can be an effective pre-natural language inference.



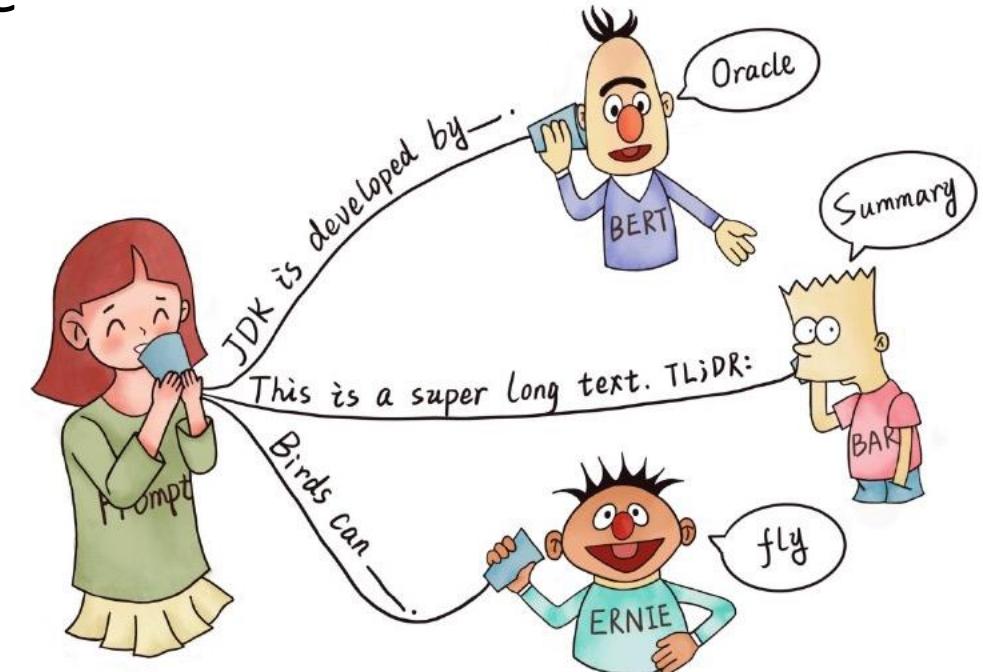
Prompting: Replacing pretraining-finetuning

- During **pas few years**, **pretraining + finetuning** becomes a paradigm for downstream tasks
- For large models, **fine-tuning** hundreds of billions of parameters (even a few iterations) is also **costly**!
- How to reduce the “usage cost” of these language models?



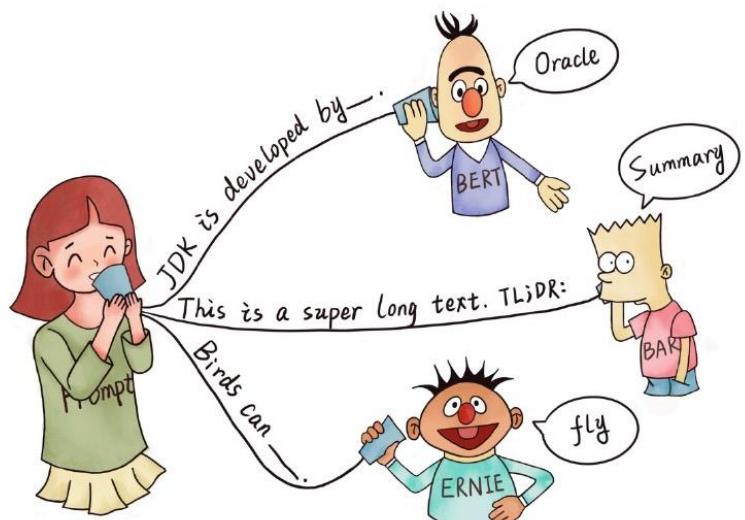
What is a prompt?

- A prompt is a **piece of text** inserted in the input examples, so that the original task can be formulated as a (masked) language modeling problem.
- For example, we want to classify the sentiment of the movie review **“No reason to watch”**, we can append a prompt **“It was”** to the sentence, getting **“No reason to watch. It was ____.”** It is natural to expect a higher probability from the language model to generate “terrible” than “great”.



Prompt engineering

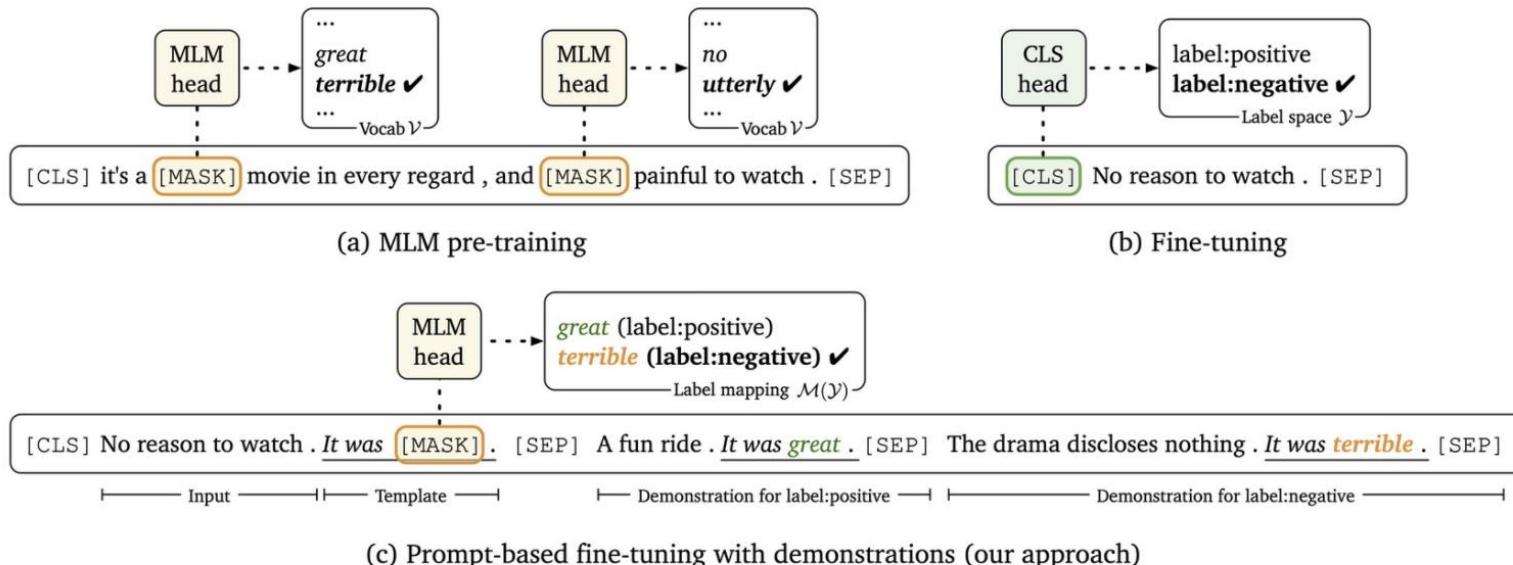
ChatGPT + Prompt



- | | |
|--|--|
|  Q&A
Answer questions based on existing knowle... |  Grammar correction
Corrects sentences into standard English. |
|  Summarize for a 2nd grader
Translates difficult text into simpler concep... |  Natural language to OpenAI API
Create code to call to the OpenAI API usin... |
|  Text to command
Translate text into programmatic commands. |  English to other languages
Translates English text into French, Spanish... |
|  Natural language to Stripe API
Create code to call the Stripe API using nat... |  SQL translate
Translate natural language to SQL queries. |
|  Parse unstructured data
Create tables from long form text |  Classification
Classify items into categories via example. |
|  Python to natural language
Explain a piece of Python code in human un... |  Movie to Emoji
Convert movie titles into emoji. |
|  Calculate Time Complexity
Find the time complexity of a function. |  Translate programming languages
Translate from one programming language ... |
|  Advanced tweet classifier
Advanced sentiment detection for a piece o... |  Explain code
Explain a complicated piece of code. |
|  Keywords
Extract keywords from a block of text. |  Factual answering
Guide the model towards factual answering ... |
|  Ad from product description
Turn a product description into ad copy. |  Product name generator
Create product names from examples word... |
|  TL;DR summarization
Summarize text by adding a 'tl;dr' to the en... |  Python bug fixer
Find and fix bugs in source code. |
|  Spreadsheet creator
Create spreadsheets of various kinds of dat... |  JavaScript helper chatbot
Message-style bot that answers JavaScript ... |
|  ML/AI language model tutor
Bot that answers questions about language... |  Science fiction book list maker
Create a list of items for a given topic. |

Why prompt learning?

- Prompting makes it possible for downstream tasks to take the same format as the pre-training objectives and **requires no new parameters**.
- Just need to **design a template** ("It was") and the expected text responses (we call these label words, e.g., "great" for the positive label and "terrible" for the negative label).
- **Sampling efficiency for few-shot** case--a dozen of training examples for a new task, it is hard to fine-tune the pre-trained models and the new task-specific parameters effectively, but the process is much smoother with prompting.



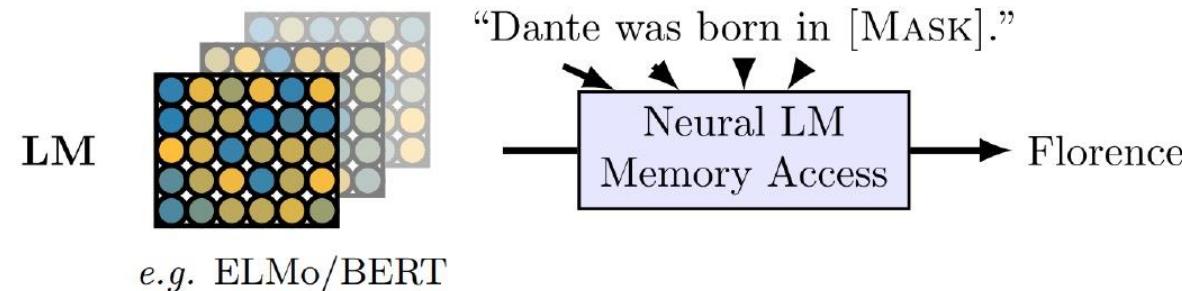
An illustration for pre-training, standard fine-tuning and prompt-based fine-tuning with demonstrations, taking a sentiment classification task as an example (from Gao et al., 2021).

Prompt engineering

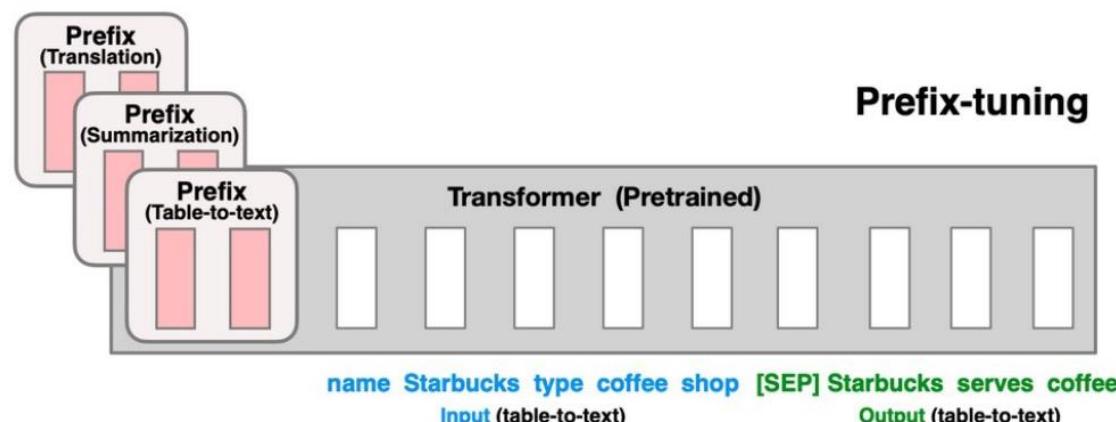
- **Pre-trained model choice**: There are a wide variety of pre-trained LMs that could be used, BERT, GPT, T5 etc.
- **Prompt engineering**: Given that the prompt specifies the task, choosing a proper prompt has a large effect not only on the accuracy, but also on which task the model performs in the first place.
- **Prompt-based training strategies**: There are also methods to train parameters, either of the prompt, the LM, or both.

Prompt shape

- **Cloze prompts:** fill in the blanks of a textual string



- **Prefix prompts:** continue a string prefix



Manual prompt engineering

- Create intuitive **templates based on human** introspection.
 - The seminal LAMA [1] dataset provides **manually created cloze templates** to probe knowledge in LMs.
 - Brown et al. [5] create **manually crafted prefix prompts** to handle a wide variety of tasks, including question answering, translation, and probing tasks for common-sense reasoning.
- Issues with manual prompt engineering
 - creating and experimenting with these prompts **is an art** that **takes time and experience**
 - even experienced prompt designers may **fail to manually discover optimal prompts**

Automated prompt searching

- **Discrete prompts** (a.k.a. **hard prompts**): templates described in a discrete space, usually corresponding to natural language phrases.
- **Continuous prompts** (a.k.a. **soft prompts**): rather than being for human consumption, it is not necessary to limit the prompt to human-interpretable natural. Continuous prompts perform prompting **directly in the embedding space** of the model.

Answer to prompts

- Language model is used to **fill the unfilled slot** to obtain final outputs.
- **What is the answer to prompts?**
- Some common choices for the shape of an answer include:
 - **Tokens**: One of the tokens in the pre-trained LM's vocabulary, or a subset of the vocabulary [1, 2].
 - **Span**: A short multi-token span. These are usually used together with cloze prompts [12].
 - **Sentence**: A sentence or document. These are commonly used with prefix prompts.

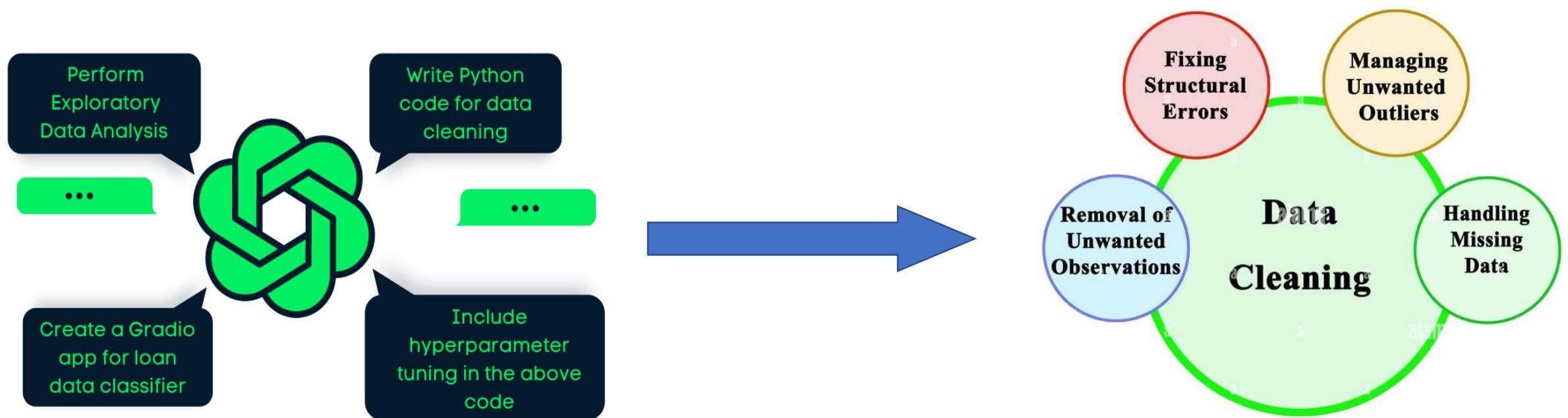
Writing effective ChatGPT prompts

Prompt engineering is tricky when it comes to using it for real-life projects. We need to understand the rules about what we can do or where we have to step in to correct the ChatGPT.

Here are some tips on how you can improve your experience with ChatGPT without compromising the project.

1. **Always write clear and concise prompts.** Make sure you explain everything in detail about what you need at the start.
2. **Create a history of a project.** ChatGPT is a chatbot, so for it to effectively understand the context, you need to establish history.
3. **Keep trying.** There is no standard way of writing prompts. You need to start with a base prompt and keep improving the suits by writing follow-up update prompts.
4. **Mention code errors.** If you run the code on your local machine and it raises the error, try mentioning the error in the follow-up prompt. ChatGPT will immediately learn from the mistake and come up with a better solution.
5. **Make changes manually.** ChatGPT was trained on old data, and if you are expecting it to come up with the idea or new API command, you will be disappointed. Try making manual changes to your code where possible because the generated code is not perfect.
6. **Use it for common tasks.** There are better chances of success with ChatGPT if you ask for common tasks.

LLMs for data preprocessing



Fetch dataset via LLMs

- Ask LLMs about open-source dataset link
- Let LLMs generate codes to obtain dataset

Give me a link to Ubuntu Dialogue dataset

Assistant [Poe](#) ...

The Ubuntu Dialogue Corpus is a widely used dataset for dialogue system research. It contains multi-turn conversations extracted from the Ubuntu chat logs, where users discuss various technical issues related to Ubuntu operating system.

You can access the Ubuntu Dialogue Corpus through the following link:

<https://github.com/rkadlec/ubuntu-ranking-dataset-creator>

This link will direct you to the GitHub repository for the "ubuntu-ranking-dataset-creator." The repository provides scripts and instructions to create the Ubuntu Dialogue Corpus from the raw Ubuntu chat logs.

Please note that the dataset is provided in a text format and requires preprocessing to convert it into a suitable format for your specific task or research. The GitHub repository contains detailed guidelines on how to preprocess the dataset and use it for dialogue system experiments.

Prompt: Today we'll perform data cleaning and preprocessing on a real-world dataset. Write the code to do the following: 1. Fetch the bank marketing dataset from the UCI machine learning repository. Use libraries like requests or urllib. 2. Download the zip file, unzip the contents, and read in the bank-full.csv file into a pandas dataframe (call it data).

The code to fetch and read in the dataset is as follows:

```
import pandas as pd
import urllib
import zipfile

# URL of the dataset
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/00222/bank.zip"

# Fetch and extract the dataset
urllib.request.urlretrieve(url, "bank.zip")

with zipfile.ZipFile("bank.zip", "r") as zip_ref:
    zip_ref.extractall()

# Load the dataset into a pandas DataFrame
data = pd.read_csv("bank-full.csv", delimiter=";", skipinitialspace=True)
```

Data preprocessing via LLMs

- Generic Data Cleaning and Preprocessing Pipeline
 - Though data cleaning and preprocessing require efforts from the developer, let's try to get a generic sequence of steps that works reasonably well for a simple dataset.
- Prompt Example
 - *Can you give me a generic data cleaning and preprocessing pipeline based on what we've done so far. Get basic info on the dataset, check for and handle missing values, identify categorical columns, and encode categorical columns. Use only pandas.*

Data preprocessing pipelines via LLMs

- Generic Data Cleaning and Preprocessing Pipeline

Can you give me a generic data cleaning and preprocessing pipeline based on what we've done so far. Get basic info on the dataset, check for and handle missing values, identify categorical columns, and encode categorical columns. Use only pandas.

Assistant [Poe](#)

Certainly! Here's a generic data cleaning and preprocessing pipeline using pandas, based on the steps you mentioned:

```
python Copy
import pandas as pd

# Load the dataset
df = pd.read_csv('your_dataset.csv')

# Get basic information on the dataset
print("Dataset Information:")
print(df.info())

# Check for missing values
print("Missing Values:")
print(df.isnull().sum())

# Handle missing values
# Example: Replace missing values with the mean of each column
df.fillna(df.mean(), inplace=True)
```

```
# Check for missing values
print("Missing Values:")
print(df.isnull().sum())

# Handle missing values
# Example: Replace missing values with the mean of each column
df.fillna(df.mean(), inplace=True)

# Identify categorical columns
categorical_columns = df.select_dtypes(include='object').columns

# Encode categorical columns using one-hot encoding
encoded_df = pd.get_dummies(df, columns=categorical_columns)

# Print the preprocessed dataset
print("Preprocessed Dataset:")
print(encoded_df)
```

This pipeline assumes that your dataset is in a CSV file format. Here's a breakdown of the steps:

1. Load the dataset using `pd.read_csv('your_dataset.csv')`. Replace 'your_dataset.csv' with the path to your actual dataset file.
2. Get basic information on the dataset using `df.info()`. This provides an overview of the column names, data types, and non-null

1. Load the dataset using `pd.read_csv('your_dataset.csv')`. Replace 'your_dataset.csv' with the path to your actual dataset file.
2. Get basic information on the dataset using `df.info()`. This provides an overview of the column names, data types, and non-null counts.
3. Check for missing values using `df.isnull().sum()`. This counts the number of missing values in each column.
4. Handle missing values according to your desired strategy. In the example, missing values are replaced with the mean of each column using `df.fillna(df.mean(), inplace=True)`. You can modify this step to handle missing values based on your specific requirements.
5. Identify categorical columns using `df.select_dtypes(include='object').columns`. This selects columns with object (string) data type as categorical columns. Modify the `include` parameter if your categorical columns have a different data type.
6. Encode categorical columns using one-hot encoding with `pd.get_dummies(df, columns=categorical_columns)`. This creates binary columns for each unique category in the categorical columns.
7. Print the preprocessed dataset using `print(encoded_df)`. This displays the dataset with the encoded categorical columns.

Handle missing values via LLMs

- Ask LLMs to provide methods to process missing values

Prompt Example:

What are some effective techniques for handling missing data in my dataset?

How can I impute missing values or deal with missing data effectively?

What are some effective techniques for handling missing data in my dataset? How can I impute missing values or deal with missing data effectively?

Assistant Poe

...

Handling missing data is an important step in data preprocessing. Here are some effective techniques for dealing with missing data:

1. Identify the Missing Data Patterns: Start by exploring your dataset to understand the patterns and extent of missing data. Identify if the missingness is random or if there is a systematic pattern to it. This information can help determine the most appropriate technique for handling missing data.
2. Deletion: If the missing data is minimal and randomly distributed, you may consider deleting the rows or columns with missing values. However, be cautious as this approach may lead to loss of valuable information if the missingness is not random.
 - Listwise Deletion: Delete entire rows with missing values.
 - Pairwise Deletion: Use available data for each specific analysis, ignoring missing values for specific variables.
3. Mean/Median/Mode Imputation: Replace missing values with the mean, median, or mode of the available values in the corresponding variable. This method assumes that the missing values are missing completely at random (MCAR) and may distort the original

Handle missing values via LLMs

- Ask ChatGPT to generate codes to implement the proposed methods

Prompt Example:

Please Generate a python code to implement the technique 3.

Please Generate a python code to implement the technique 3.

Assistant Poe

Certainly! Here's an example Python code to implement Technique 3, which is mean imputation for handling missing data:

```
python
import pandas as pd
import numpy as np

# Load your dataset
df = pd.read_csv('your_dataset.csv')

# Check for missing values
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values)

# Perform mean imputation
for column in df.columns:
    if df[column].isnull().any():
        mean_value = df[column].mean()
        df[column].fillna(mean_value, inplace=True)

# Verify if missing values are imputed
missing_values_after_imputation = df.isnull().sum()
print("Missing Values After Imputation:\n", missing_values_after_imputation)
```

Remove outliers via LLMs

- Seek guidance from LLMs on outlier detection methods and techniques for removing outliers from your dataset.

What are some effective techniques for handling missing data in my dataset? How can I impute missing values or deal with missing data effectively?

...

Assistant Poe

Handling missing data is an important step in data preprocessing. Here are some effective techniques for dealing with missing data and imputing missing values:

1. Deletion:

- Listwise Deletion: Delete entire rows with missing values. This approach is straightforward but may result in a loss of valuable data.
- Pairwise Deletion: Use available data for each specific analysis, ignoring missing values for specific variables. This approach allows you to retain more data but can lead to biased results if missingness is not random.

2. Mean/Median/Mode Imputation:

- Mean Imputation: Replace missing values with the mean of the available values in the corresponding variable. It assumes that the missing values are missing completely at random (MCAR) and may distort the original distribution of the data.
- Median Imputation: Similar to mean imputation, but replace

• Mean Imputation: Replace missing values with the mean of the available values in the corresponding variable. It assumes that the missing values are missing completely at random (MCAR) and may distort the original distribution of the data.

• Median Imputation: Similar to mean imputation, but replace missing values with the median instead. It can be more robust to outliers.

• Mode Imputation: Replace missing values with the mode (most frequent value) of the variable. It is suitable for categorical or discrete variables.

3. Regression Imputation:

• Predictive Mean Matching: Fit a regression model with the variable containing missing values as the dependent variable and the other variables as predictors. Then use the predicted values from the regression model to impute the missing values.

• Multiple Imputation by Regression: Generate multiple imputations using regression models and combine the results. This technique accounts for the uncertainty in imputing missing values.

Remove outliers via LLMs

- Ask ChatGPT to generate codes to implement the above methods

Prompt Example

Please generate python codes to implement the all above methods and users could choose one method to use.

Please generate python codes to implement the all above methods and users could choose one method to use.

Assistant Poe

...

Certainly! Here's an example Python code that implements different methods for handling missing data. Users can choose the desired method by commenting/uncommenting the corresponding sections:

```
python
import pandas as pd
import numpy as np
from sklearn.impute import KNNImputer

# Load your dataset
df = pd.read_csv('your_dataset.csv')

# Check for missing values
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values)

# Method 1: Deletion
# df.dropna(inplace=True)
# print("Dataset after Deletion:\n", df)
```

```
# Method 2: Mean/Median/Mode Imputation
# df.fillna(df.mean(), inplace=True)
# df.fillna(df.median(), inplace=True)
# df.fillna(df.mode().iloc[0], inplace=True)
# print("Dataset after Mean/Median/Mode Imputation:\n", df)

# Method 3: Regression Imputation
# Perform regression imputation for each column with missing values
# for column in df.columns:
#     if df[column].isnull().any():
#         observed = df.loc[~df[column].isnull()]
#         missing = df.loc[df[column].isnull()]
#         X = observed.drop(column, axis=1)
#         y = observed[column]
#         # Perform regression to predict missing values
#         regression_model = YourRegressionModel()
#         regression_model.fit(X, y)
#         imputed_values = regression_model.predict(missing.drop(column, axis=1))
#         df.loc[df[column].isnull(), column] = imputed_values
# print("Dataset after Regression Imputation:\n", df)

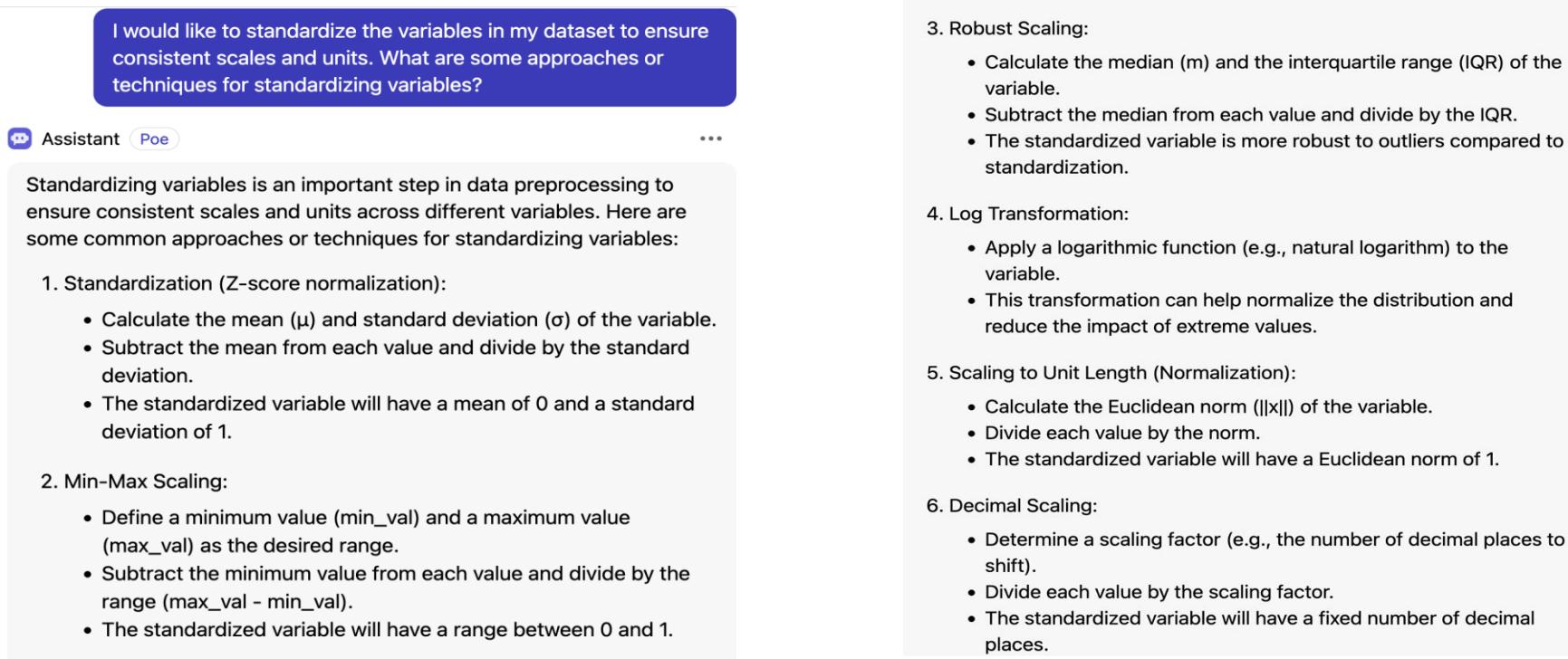
# Method 4: Hot Deck Imputation
# Perform hot deck imputation using random or deterministic hot dec
```

Standardize variables via LLMs

- Ask ChatGPT to give techniques and implement codes

Prompt Example:

I would like to standardize the variables in my dataset to ensure consistent scales and units. What are some approaches or techniques for standardizing variables?



I would like to standardize the variables in my dataset to ensure consistent scales and units. What are some approaches or techniques for standardizing variables?

Assistant Poe

Standardizing variables is an important step in data preprocessing to ensure consistent scales and units across different variables. Here are some common approaches or techniques for standardizing variables:

1. Standardization (Z-score normalization):
 - Calculate the mean (μ) and standard deviation (σ) of the variable.
 - Subtract the mean from each value and divide by the standard deviation.
 - The standardized variable will have a mean of 0 and a standard deviation of 1.
2. Min-Max Scaling:
 - Define a minimum value (min_val) and a maximum value (max_val) as the desired range.
 - Subtract the minimum value from each value and divide by the range (max_val - min_val).
 - The standardized variable will have a range between 0 and 1.

3. Robust Scaling:

- Calculate the median (m) and the interquartile range (IQR) of the variable.
- Subtract the median from each value and divide by the IQR.
- The standardized variable is more robust to outliers compared to standardization.

4. Log Transformation:

- Apply a logarithmic function (e.g., natural logarithm) to the variable.
- This transformation can help normalize the distribution and reduce the impact of extreme values.

5. Scaling to Unit Length (Normalization):

- Calculate the Euclidean norm ($\|x\|$) of the variable.
- Divide each value by the norm.
- The standardized variable will have a Euclidean norm of 1.

6. Decimal Scaling:

- Determine a scaling factor (e.g., the number of decimal places to shift).
- Divide each value by the scaling factor.
- The standardized variable will have a fixed number of decimal places.

Standardize variables via LLMs

7. Feature Scaling with Machine Learning:

- Use machine learning algorithms that automatically scale the variables.
- Examples include algorithms like Support Vector Machines (SVM) and K-Nearest Neighbors (KNN).

When choosing an approach for standardizing variables, consider the characteristics of your data and the requirements of your analysis. Some methods may be more suitable for certain types of data or specific machine learning algorithms. It's also important to note that standardization should be applied after handling missing values or outliers, if applicable.

Implementing these approaches in Python is relatively straightforward. You can use libraries such as scikit-learn (sklearn) or pandas (pd) to apply the transformations. Here's an example using scikit-learn's `StandardScaler` for Z-score normalization:

```
python
from sklearn.preprocessing import StandardScaler

# Load your dataset
df = pd.read_csv('your_dataset.csv')
```

python

Copy

```
from sklearn.preprocessing import StandardScaler

# Load your dataset
df = pd.read_csv('your_dataset.csv')

# Select the variables to be standardized
variables_to_standardize = ['variable1', 'variable2', 'variable3']

# Create a StandardScaler object
scaler = StandardScaler()

# Fit the scaler on the selected variables
scaler.fit(df[variables_to_standardize])

# Transform the selected variables using the scaler
df[variables_to_standardize] = scaler.transform(df[variables_to_standardize])

# Print the standardized dataset
print("Standardized Dataset:\n", df)
```

This code snippet demonstrates how to standardize selected variables using the `StandardScaler` from scikit-learn. You can modify it to include other standardization techniques or adapt it to your specific requirements.

Remember to apply the same transformation to new data or test sets using the fitted scaler to ensure consistent scaling.

Categorical variables transformation via LLMs

Prompt Example:

I have categorical variables in my dataset. Could you recommend methods for encoding or transforming categorical variables into numerical representations for analysis?

Assistant Poe

Certainly! Encoding categorical variables into numerical representations is a common step in data preprocessing to enable analysis with machine learning algorithms. Here are some popular methods for encoding categorical variables:

1. One-Hot Encoding:

- Create binary columns for each unique category in the variable.
- Assign a value of 1 to the corresponding category and 0 to the rest.
- This method is suitable for nominal variables (categories with no intrinsic ordering).

2. Label Encoding:

- Assign a unique numerical label to each category.
- The labels are assigned based on the order of appearance or some specific criteria.
- This method is suitable for ordinal variables (categories with a natural ordering).

Stop

3. Ordinal Encoding:

- Assign numerical values to categories based on their order or predefined mapping.
- The values are assigned in a way that preserves the ordinal relationship between categories.
- This method is suitable for ordinal variables.

4. Count Encoding:

- Replace categories with the count of their occurrences in the dataset.
- This method can capture the frequency information of each category.

5. Target Encoding:

- Replace categories with the mean (or other aggregated statistic) of the target variable for each category.
- This method can capture the relationship between the categorical variable and the target variable.

6. Binary Encoding:

- Represent each category with binary digits.
- Each category is assigned a unique binary code, and the codes are used as numerical representations.
- This method can be efficient for high-cardinality categorical

Categorical variables transformation via LLMs

6. Binary Encoding:

- Represent each category with binary digits.
- Each category is assigned a unique binary code, and the codes are used as numerical representations.
- This method can be efficient for high-cardinality categorical variables.

7. Feature Hashing:

- Apply a hashing function to transform categories into numerical representations.
- The output can be a fixed-length vector, which reduces dimensionality.
- This method is useful when dealing with a large number of categories.

8. Entity Embedding:

- Represent each category as a dense vector in a lower-dimensional space.
- The embeddings are learned through a neural network model that considers the relationships between categories.
- This method is commonly used for categorical variables with high cardinality.

The choice of encoding method depends on the nature of your

The choice of encoding method depends on the nature of your categorical variables, the number of categories, and the requirements of your analysis. It's important to consider the implications of each method and how they may affect your analysis. Additionally, some machine learning algorithms can handle categorical variables directly, so it's worth checking the requirements of the specific algorithm you plan to use.

Implementing these encoding methods in Python can be done using various libraries. Here's an example using scikit-learn's OneHotEncoder for one-hot encoding:

```
python Copy
from sklearn.preprocessing import OneHotEncoder

# Load your dataset
df = pd.read_csv('your_dataset.csv')

# Select the categorical variables to be one-hot encoded
categorical_variables = ['variable1', 'variable2', 'variable3']

# Create a OneHotEncoder object
encoder = OneHotEncoder(drop='first', sparse=False)

# Fit the encoder on the selected variables
encoder.fit(df[categorical_variables])

# Transform the selected variables using the encoder
encoded_variables = encoder.transform(df[categorical_variables])
```

```
# Fit the encoder on the selected variables
encoder.fit(df[categorical_variables])

# Transform the selected variables using the encoder
encoded_variables = encoder.transform(df[categorical_variables])

# Create a DataFrame with the encoded variables
encoded_df = pd.DataFrame(encoded_variables, columns=encoder.get_feature_names(categorical_variables))

# Replace the original categorical variables with the encoded variables
df = pd.concat([df.drop(categorical_variables, axis=1), encoded_df], axis=1)

# Print the dataset with encoded variables
print("Dataset with Encoded Variables:\n", df)
```

This code snippet demonstrates how to one-hot encode categorical variables using the OneHotEncoder from scikit-learn. You can adapt it to include other encoding methods or use other libraries like pandas (`pd.get_dummies()`) for one-hot encoding.

Remember to consider the potential issue of high-dimensional feature spaces when using one-hot encoding, especially for categorical variables with many categories. Feature selection or dimensionality reduction techniques may be necessary in such cases.

Prompts for data preprocessing

- Below is a list of 15 ChatGPT prompts for “data preparation”.
 - What steps should I follow to clean and preprocess my raw data before analysis?
 - How can I handle missing values in my dataset? Are there any imputation techniques you recommend?
 - Can you explain the concept of feature scaling and suggest methods for scaling my numerical variables?
 - Are there any outlier detection and removal techniques that I should consider during data preparation?
 - What strategies can I use to handle categorical variables? Should I perform one-hot encoding or use other approaches?

Prompts for data preprocessing

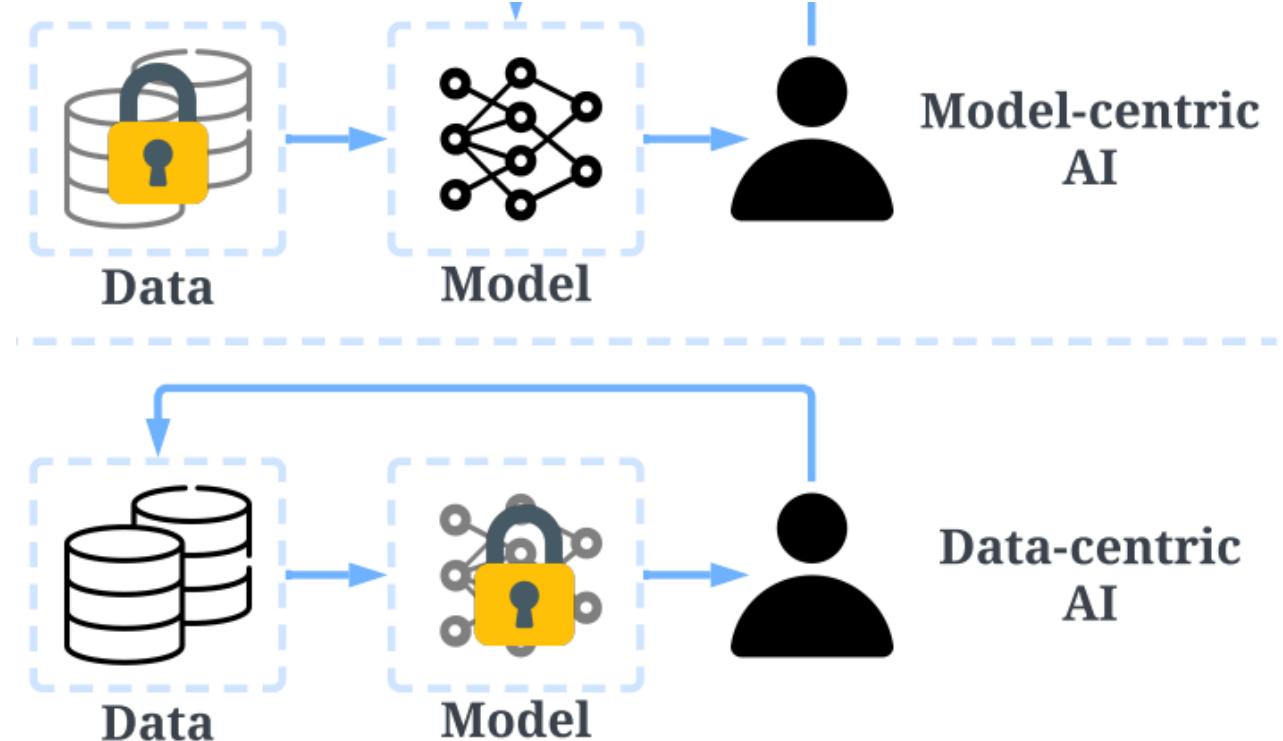
- Below is a list of 15 ChatGPT prompts for “data preparation”.
 - How do I deal with skewed distributions in my dataset? Are there any transformations that can help?
 - What are some techniques for handling multicollinearity among features in data preparation?
 - Should I remove redundant features from my dataset? If so, what criteria should I use for feature selection?
 - How can I handle date and time variables in my dataset? Are there any specific considerations for analysis?
 - Can you explain the concept of data normalization and suggest normalization techniques for my features?

Prompts for data preprocessing

- Below is a list of 15 ChatGPT prompts for “data preparation”.
 - Are there any methods for handling text data in data preparation? How can I convert text into numerical representations?
 - Can you provide guidance on splitting my dataset into training, validation, and testing sets? What is the recommended ratio?
 - How can I address data quality issues, such as duplicates or inconsistent formatting, during data preparation?
 - What are some common data validation techniques I can use to ensure the integrity of my prepared dataset?
 - Can you suggest methods for handling class imbalance in my dataset? How can I ensure balanced training data?

Data-Centric LLMs vs Model-centric AI

- Data-centric AI emphasizes the systematic engineering of data to build AI systems, shifting our focus from model to data.
- Data-centric AI ≠ Data-Driven: Data-driven only emphasizes the use of data to guide AI development, still centers on developing models rather than engineering data.



LLM augmented data

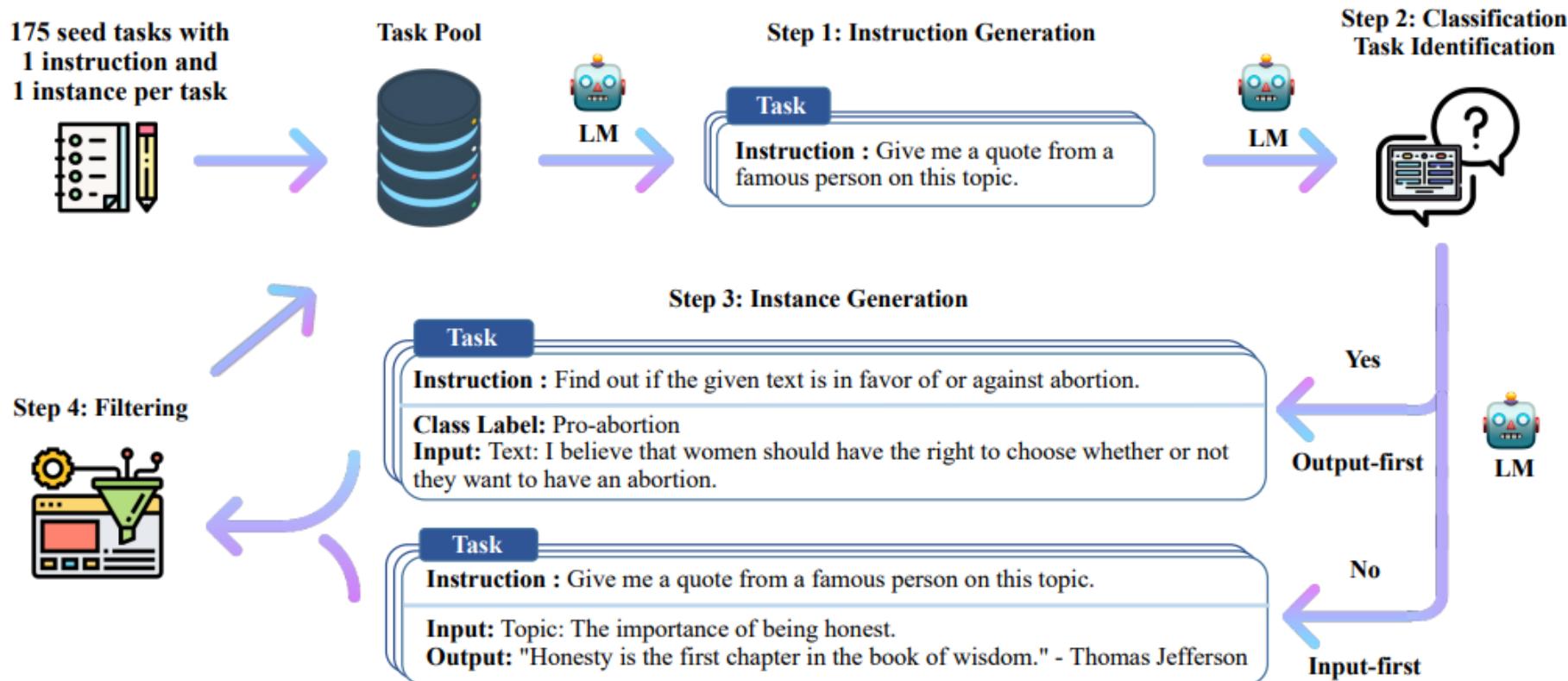
- Bottleneck of Data-Centric LLM:
 - Where does the data come from? Is the data in the world unlimited?
 - High-quality data is scarce, high-quality data are books and scientific papers, or well-annotated corpus. Low-quality data, such as chat records, phone calls, etc., have a relatively limited effect on training due to poor quality.
 - Limitations brought by data security and privacy: sensitive information such as personally identifiable information, financial information, or medical records are unavailable for LLMs.

Possible Solution: LLM-synthetic / augmented data can meet the needs.

LLM-guided data Collection

- **Model distillation:** Use the output / logits of the LLM as a label to train small models. This is often better than training a small model directly on the original dataset.
- **Self-Instruct:** Instruction-finetune that using instruction generated by LLM itself. The process is an iterative bootstrapping algorithm.

Self Instruct



LLMs for data visualization

- Visualization Code = LLM(Visualization requirement; Data description)
- Harness the power of ChatGPT to simplify our Python data visualizations!



Generate Data

- Generating Datasets with ChatGPT
- Prompt to ChatGPT: *please create as dataset for ...*

Please create a dataset for the monthly sightings of 10 different bird species in a wildlife reserve over a 12-month period. Then save it as a CSV file.

ChatGPT Poe

Certainly! Here's an example dataset for the monthly sightings of 10 different bird species in a wildlife reserve over a 12-month period:

apache

Copy

```
Month,Bird Species 1,Bird Species 2,Bird Species 3,Bird Species 4,Bird Species 5,Bird Species 6,Bird Species 7,Bird Species 8,Bird Species 9,Bird Species 10
January,15,8,3,2,10,12,5,7,9,6
February,18,9,4,3,11,14,6,8,11,7
March,20,10,5,4,12,16,8,9,13,8
April,22,12,6,5,13,18,10,11,15,9
May,25,14,7,6,15,20,12,13,17,10
June,27,15,8,7,16,22,14,15,19,11
July,30,17,9,8,17,24,16,17,21,12
August,28,16,8,7,16,23,15,16,20,11
September,26,15,7,6,15,21,14,15,19,10
October,23,13,6,5,14,19,12,13,17,9
November,19,10,4,3,12,15,8,9,13,7
December,16,9,3,2,11,13,6,8,11,6
```

To save this dataset as a CSV file, you can follow these steps:

1. Copy the dataset provided above and paste it into a plain text editor (e.g., Notepad).
2. Save the file with a ".csv" extension (e.g., "bird_sightings.csv").
Make sure to choose "All Files" as the file type when saving.

Grouped Bar Chart

Please provide Python code to create a grouped bar chart for the bird sightings dataset in 'bird_data.csv'.

ChatGPT Poe

Certainly! Here's an example Python code that uses the pandas and matplotlib libraries to create a grouped bar chart for the bird sightings dataset in the 'bird_data.csv' file:

python

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Read the CSV file into a pandas DataFrame
df = pd.read_csv('bird_data.csv')

# Set the bird species as the x-axis labels
bird_species = df.columns[1:]

# Get the total sightings for each month
total_sightings_by_month = df.sum(axis=1, numeric_only=True)

# Create an array of month labels
months = df['Month']

# Set the width of each bar
bar_width = 0.15

# Set the position of each bar on the x-axis
r = np.arange(len(months))

# Create a grouped bar chart
plt.figure(figsize=(10, 6))
for i, species in enumerate(bird_species):
    plt.bar(r + i * bar_width, df[species], width=bar_width, label=species)

# Set x-axis labels, ticks, and title
plt.xlabel('Month')
plt.xticks(r + (len(bird_species) / 2) * bar_width, months)
plt.ylabel('Sightings')
plt.title('Bird Sightings by Month and Species')

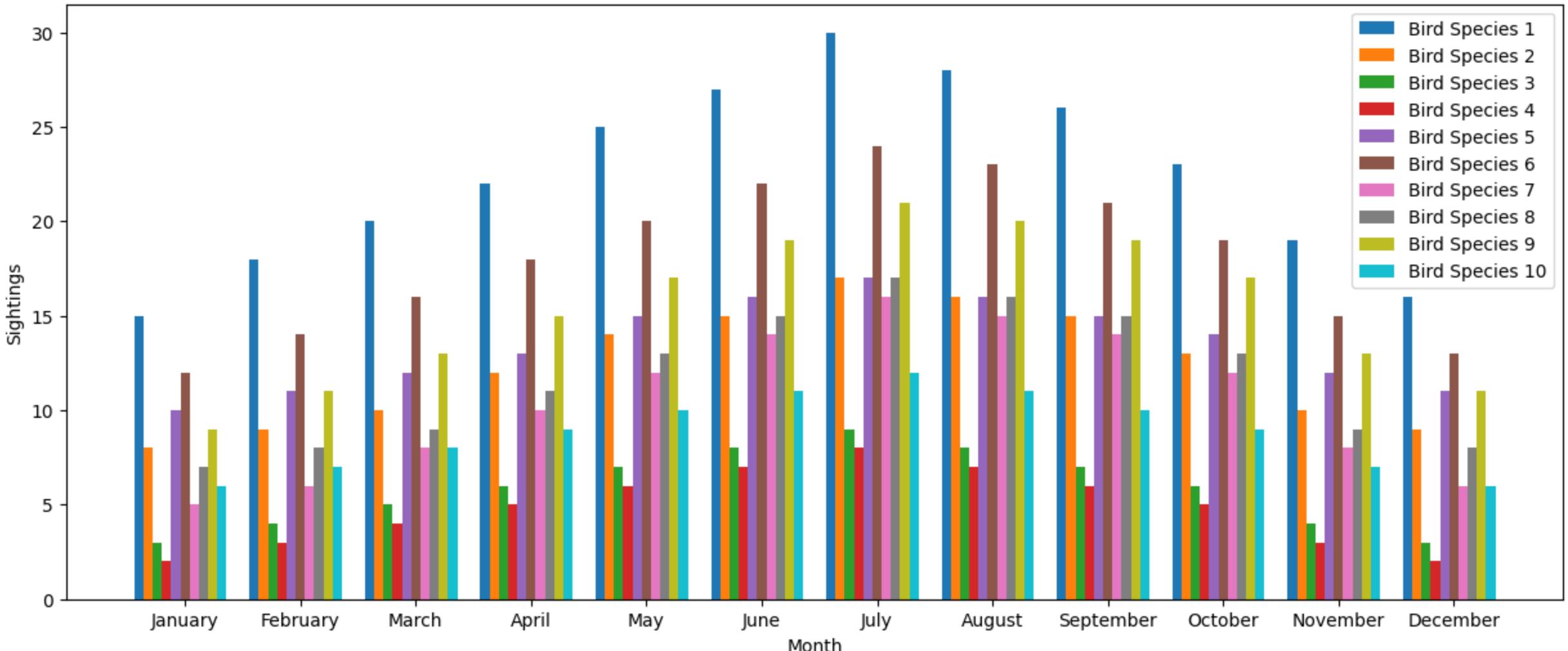
# Add a legend
plt.legend()

# Display the grouped bar chart
plt.show()
```

Copy

Grouped Bar Chart

Bird Sightings by Month and Species



Heat Map

Please provide Python code to create a heat map for the bird sightings dataset in 'bird_data.csv'.

ChatGPT Poe

Certainly! Here's an example Python code that uses the pandas, matplotlib, and seaborn libraries to create a heat map for the bird sightings dataset in the 'bird_data.csv' file:

python

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Read the CSV file into a pandas DataFrame
df = pd.read_csv('bird_data.csv', index_col='Month')

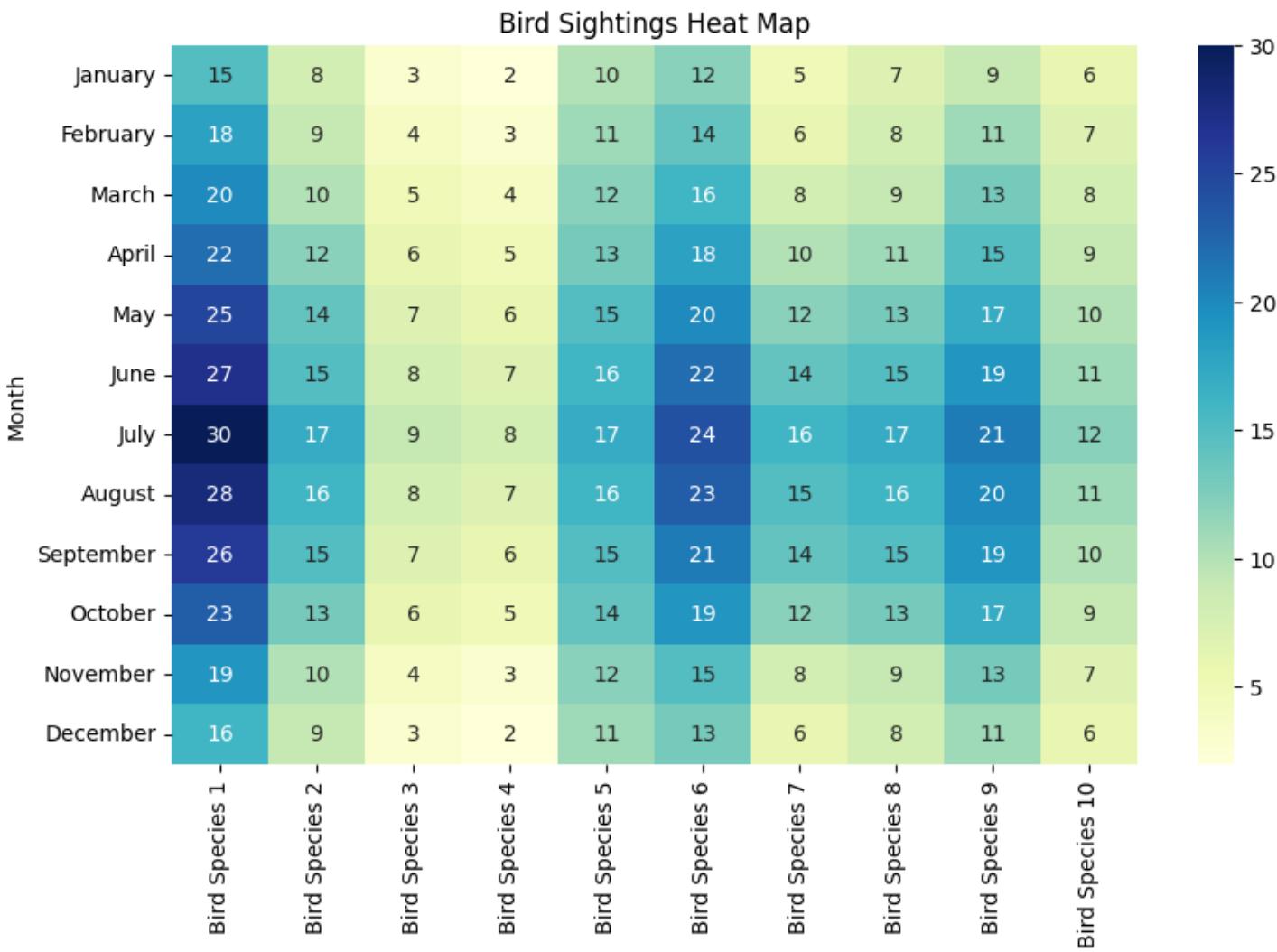
# Create a heat map
plt.figure(figsize=(10, 6))
sns.heatmap(df, cmap='YlGnBu', annot=True, fmt='d', cbar=True)

# Set the title
plt.title('Bird Sightings Heat Map')

# Display the heat map
plt.show()
```

Copy

Heat Map



Line Chart

Please provide Python code to create a line chart for the bird sightings dataset in 'bird_data.csv'.

ChatGPT Poe

Certainly! Here's the Python code to create a line chart for the bird sightings dataset in 'bird_data.csv':

python

```
import pandas as pd
import matplotlib.pyplot as plt

# Read the CSV file into a pandas DataFrame
df = pd.read_csv('bird_data.csv')

# Extract the necessary columns
months = df['Month']
bird_species = df.drop('Month', axis=1)

# Create the line chart
plt.figure(figsize=(12, 6))
for bird in bird_species.columns:
    plt.plot(months, bird_species[bird], marker='o', label=bird)

# Set the title and labels
plt.title('Bird Sightings Over Time')
plt.xlabel('Month')
plt.ylabel('Sightings')

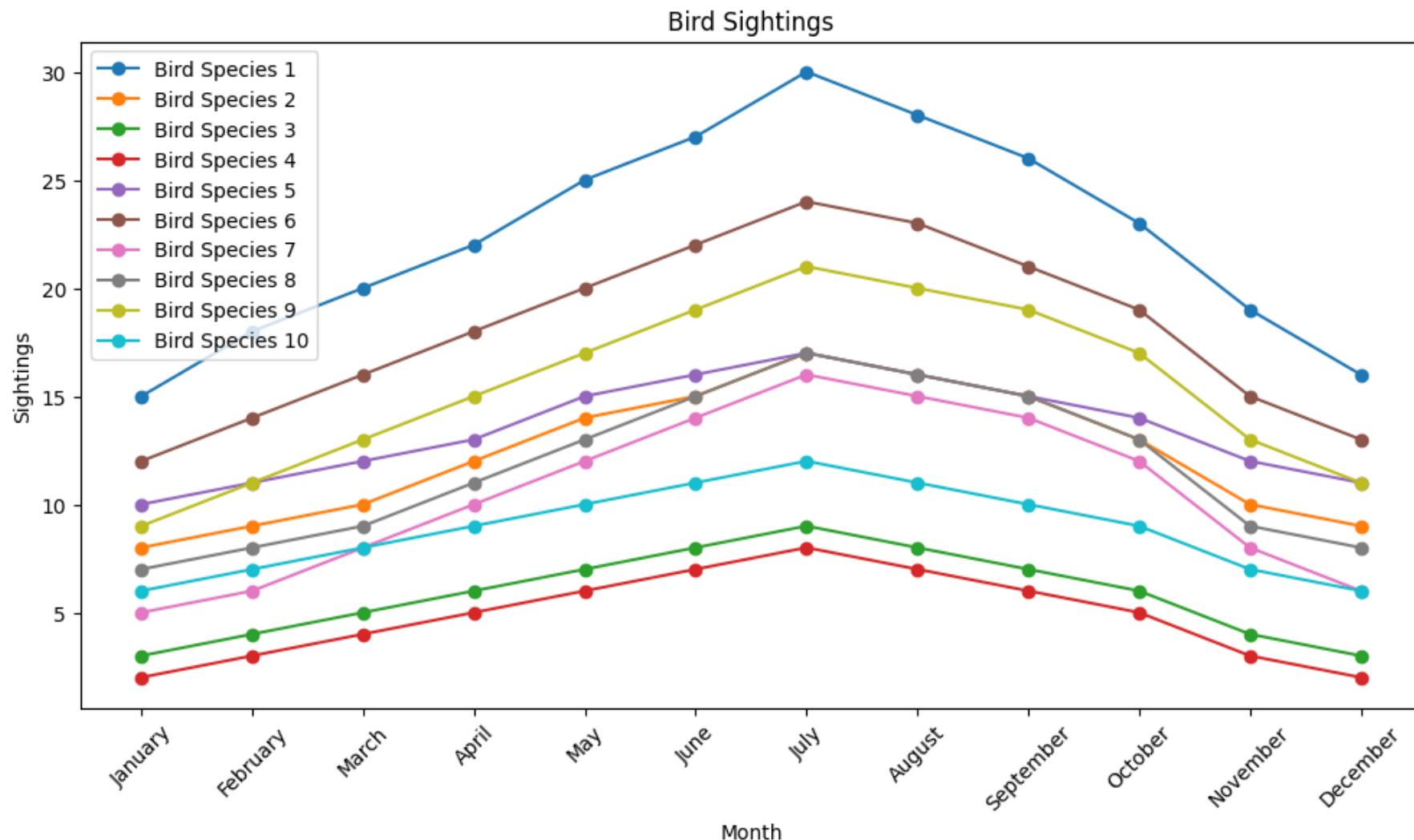
# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Add a legend
plt.legend()

# Display the chart
plt.show()
```

Copy

Line Chart



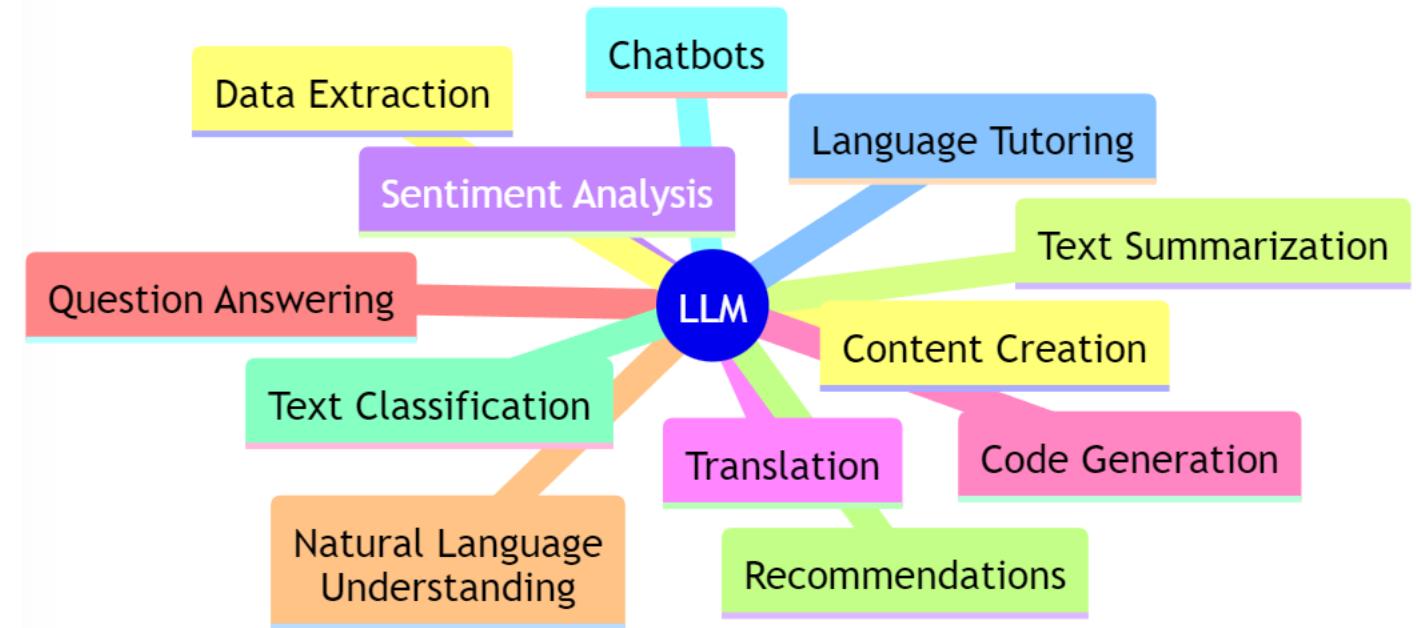
Summary: ChatGPT for data visualization

- Visualization Code = LLM(Visualization requirement; Data description)



Large Language Models for Text Analysis

- sentiment analysis
- text classification
- information extraction
- machine translation



Sentiment Analysis - Positive

I had an amazing vacation in Hawaii!

 ChatGPT 

That's wonderful to hear! Hawaii is known for its stunning natural beauty, warm beaches, and vibrant culture. I'm glad you had an amazing vacation there. Could you please share more about your experience? What were some of the highlights of your trip?

ChatGPT recognizes the **positive sentiment** expressed by the user regarding their vacation and **responds with enthusiasm, prompting for more details.**

Sentiment Analysis - Negative

I'm really disappointed with the service at this restaurant.

ChatGPT

Poe

I'm sorry to hear that you had a disappointing experience at the restaurant. It's never pleasant when expectations are not met, especially when it comes to service. Could you please provide more details about what specifically disappointed you? Understanding the specific aspects of your experience can help address the issue and provide a better response.

ChatGPT detects the **negative sentiment** expressed by the user and **responds empathetically, seeking further information** to understand the specific issues with the restaurant's service.

Sentiment Analysis - Mixed

The movie had great visuals, but the plot was disappointing.



ChatGPT

Poe

It's common to have mixed feelings about a movie when the visuals are impressive, but the plot doesn't meet expectations. A visually stunning film can captivate the audience with its aesthetics, special effects, and cinematography. However, a weak or disappointing plot can leave viewers feeling unsatisfied or disconnected from the story.

A well-crafted plot is crucial for engaging the audience, building tension, and creating a meaningful narrative. It should have compelling characters, a coherent storyline, and satisfying story arcs. When the plot falls short, it can detr

ChatGPT identifies the **mixed sentiment** expressed by the user, **acknowledging the positive aspect of the movie's visuals while addressing the disappointment regarding the plot**. It then prompts for more details about the impressive visuals.

Text Classification- Topic

Can you tell me about the latest advancements in artificial intelligence?

 ChatGPT 

As of my knowledge cutoff in September 2021, there have been several significant advancements in the field of artificial intelligence (AI). Here are a few notable areas of progress:

ChatGPT recognizes the **topic of the user's query** as "advancements in artificial intelligence" and responds by acknowledging the request and offering to provide relevant information.

Text Classification- Intent

How do I cancel my subscription?

 ChatGPT 

I'm sorry, but as an AI language model, I don't have access to personal account information or the ability to cancel subscriptions on your behalf. To cancel a subscription, you typically need to follow the specific cancellation process provided by the service or platform you subscribed to.

Here are some general steps you can take to cancel a subscription:

1. Review the Terms and Conditions: Check the terms and conditions or the documentation you received when you initially subscribed to understand the cancellation policy and any requirements.
2. Visit the Subscription Platform: Log in to the platform or website where you initially subscribed to the service. Look for an account settings or subscription management section.
3. Locate the Subscription: Find the specific subscription you want

ChatGPT recognizes the **user's intent to cancel a subscription** and provides guidance on the necessary steps to do so.

Text Classification- Product

I'm looking for a new smartphone with a good camera.

 ChatGPT 

If you're searching for a new smartphone with a good camera, there are several options to consider. Here are a few popular smartphones known for their camera capabilities as of my knowledge cutoff in September 2021:

1. iPhone 13 Pro / iPhone 13 Pro Max: Apple's flagship iPhone models are renowned for their camera quality. They feature advanced camera systems with larger sensors, improved low-light performance, and enhanced computational photography capabilities.
2. Samsung Galaxy S21 Ultra: The Galaxy S21 Ultra stands out for its impressive camera setup. It offers a high-resolution primary camera, a periscope telephoto lens for optical zoom, and advanced features like 100x Space Zoom and 8K video recording.
3. Google Pixel 6 Pro: The Pixel series has gained recognition for

ChatGPT identifies the **user's query as a request for a smartphone** recommendation with a specific feature (good camera) and responds by acknowledging the request and offering assistance.

Information Extraction - Date

When is the next public holiday in the United States?

 ChatGPT 

The next public holiday in the United States after my knowledge cutoff in September 2021 is typically Thanksgiving Day. Thanksgiving is observed on the fourth Thursday of November each year. In 2021, Thanksgiving Day falls on November 25th.

Please note that public holidays can vary by state, and there might be additional regional or state-specific holidays that are not observed nationwide. It's always a good idea to double-check with local sources or official government websites for the most accurate and up-to-date information on public holidays in your specific location.

ChatGPT extracts the relevant information (**next public holiday**) from the user's query and **provides the specific date and details.**

Information Extraction - Number

How many episodes are there in the first season of Game of Thrones?

 ChatGPT Poe

The first season of Game of Thrones, based on the "A Song of Ice and Fire" novels by George R.R. Martin, consists of 10 episodes. These episodes introduce viewers to the complex world of Westeros and its various noble houses, political scheming, and the looming threat beyond the Wall. Game of Thrones quickly gained widespread popularity and acclaim for its intricate storytelling, compelling characters, and unexpected plot twists.

ChatGPT extracts the specific information (**number of episodes**) from the user's query and provides the accurate answer.

Information Extraction - Name

Who directed the movie Inception?

 ChatGPT 

The movie Inception was directed by Christopher Nolan. Released in 2010, Inception is a science fiction heist thriller that delves into the world of dreams and subconscious manipulation. Christopher Nolan, known for his distinctive storytelling style and complex narratives, both directed and wrote the screenplay for Inception. The film features a star-studded cast, including Leonardo DiCaprio, Joseph Gordon-Levitt, Ellen Page, Tom Hardy, and Marion Cotillard, among others. Inception received critical acclaim for its innovative concept, visual effects, and thought-provoking storyline.

ChatGPT identifies the movie name (Inception) in the user's query and extracts the relevant information (**director's name**) to provide a concise answer.

Summarization – Book Summary

Can you summarize the plot of the novel "To Kill a Mockingbird" in 3 sentences.

 ChatGPT Poe

"To Kill a Mockingbird" is a novel set in the 1930s in Maycomb, Alabama. The story follows Scout Finch, her brother Jem, and their father Atticus, who defends a black man, Tom Robinson, accused of rape. Through the eyes of Scout, the book explores themes of racial prejudice, social injustice, and the loss of innocence in a small Southern town.

ChatGPT generates a summary of the plot for the novel "To Kill a Mockingbird," **capturing the essential elements and themes of the story.**

Translation

Can you translate "Hello, how are you?" to French?

 ChatGPT 

Certainly! "Hello, how are you?" can be translated to French as "Bonjour, comment ça va ?"

Please translate the sentence "Where is the nearest train station?" to German.

 ChatGPT 

The sentence "Where is the nearest train station?" can be translated to German as "Wo ist der nächste Bahnhof?"

Can you help me translate this paragraph about technology advancements to Spanish?

 ChatGPT 

Of course! I'd be happy to help you translate your paragraph about technology advancements to Spanish. Please provide the paragraph, and I'll assist you with the translation.

Challenges

- Prompting with structured information:
 - How to express tree, graph, table or relational structures in prompt learning is a major challenge.
- Entanglement of template and answer:
 - The performance highly depend on both the templates and answer mapping. How to effectively search or learn for the best combination of template and answer mappings is still a challenge.

Reference (1)

- [1] Fabio Petroni, Tim RocktÅnaschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2463–2473, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [2] Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. Template-based named entity recognition using BART. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 1835–1845, Online, August 2021. Association for Computational Linguistics.
- [3] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4582–4597, Online, August 2021. Association for Computational Linguistics.
- [4] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
- [5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- [6] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How Can We Know What Language Models Know? Transactions of the Association for Computational Linguistics, 8:423–438, 07 2020.
- [7] Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing NLP. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2153–2162, Hong Kong, China, November 2019. Association for Computational Linguistics.

Reference (2)

- [8] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 3816–3830, Online, August 2021. Association for Computational Linguistics.
- [9] Joe Davison, Joshua Feldman, and Alexander Rush. Commonsense knowledge mining from pretrained models. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 1173–1178, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [10] Taylor Shin, Yasaman Razeghi, Robert L. Logan, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts, 2020.
- [11] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks, 2021.
- [12] Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. X-FACTR: Multilingual factual knowledge retrieval from pretrained language models. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 5943–5959, Online, November 2020. Association for Computational Linguistics.
- [13] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- [14] Wenpeng Yin, Jamaal Hay, and Dan Roth. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3914–3923, Hong Kong, China, November 2019. Association for Computational Linguistics.