Pearson

LLMs For Data Science

https://github.com/DataForScience/LLMsForDataScience

Bruno Gonçalves

Data For Science, Inc

www.data4sci.com/newsletter

data4sci.substack.com

Contacts

Bruno Gonçalves

https://data4sci.com

info@data4sci.com

https://data4sci.com/call

https://www.linkedin.com/in/bmtgoncalves/

Bruno Gonçalves is an author, public speaker, corporate trainer, and consultant specializing in Generative AI, Blockchain Analytics, and Machine Learning. He founded Data For Science, Inc in 2009 to help individuals and companies solve their data driven problems.

Table of Contents:

1. Generative AI for Data Science
2. Prompt Engineering for DS
3. NLP with HuggingFace
4. Text to Speech with OpenAI

1. Generative AI for Data Science

Generative AI

- Any algorithm that can create new content (text, images, sound, etc) based on training data.
- Common approaches:
  - Generative Adversarial Networks (GANs)
  - Variational Autoencoders (VAEs)
  - Transformers
- Applications:
  - Text Generation: ChatGPT, Claude, LLama, etc
  - Image Generation: DALL-E, Midjourney, etc
  - Audio: Whisper, Conformer, Wav2Vec, etc
  - Synthetic Data: ChatGPT, BERT, etc
  - Code Generation: GitHub Copilot, Cursor, etc

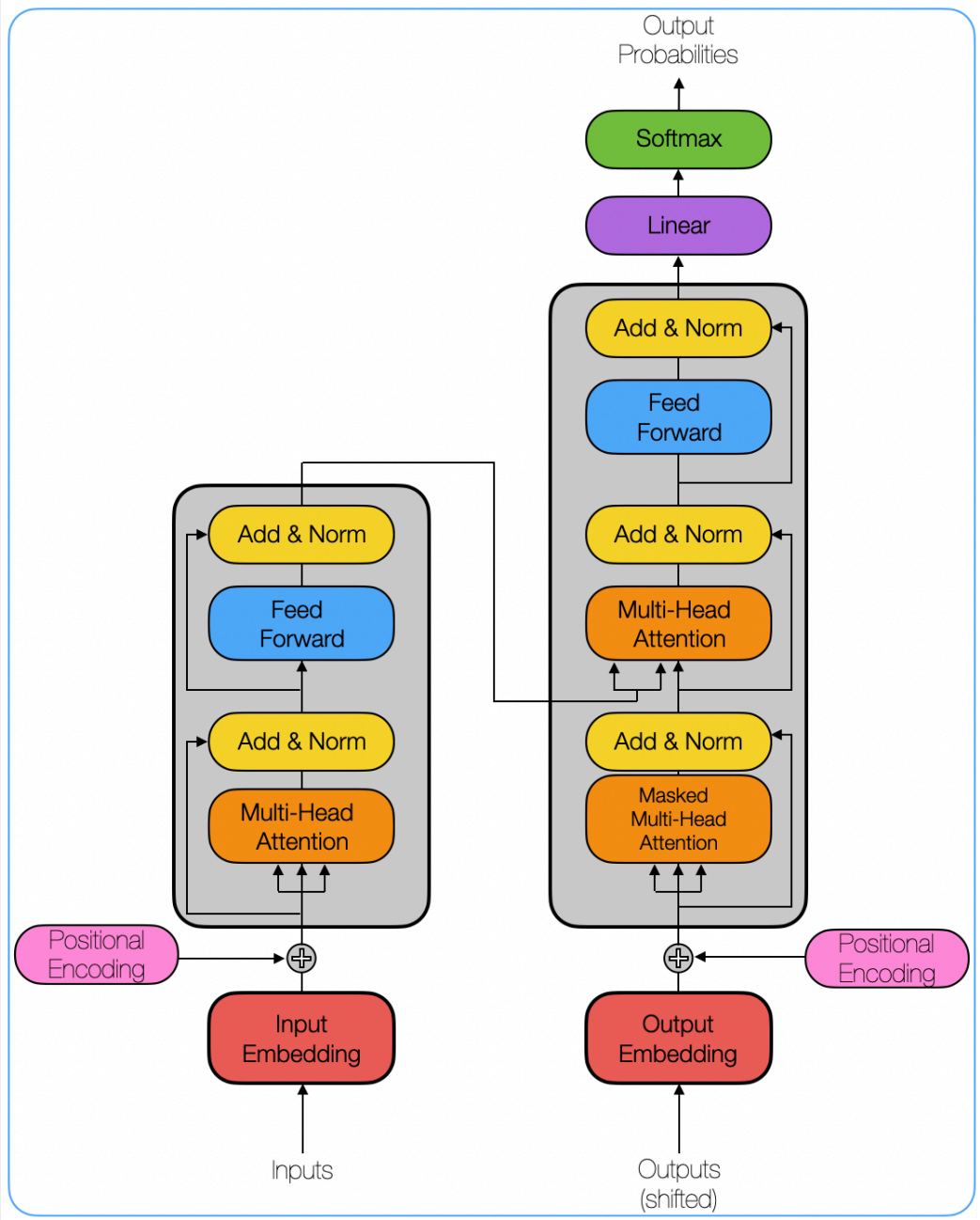
Language Models

- A foundational component of Natural Language Processing and AI.
- Software that predicts and generates human language based on patterns observed in training data.
- Applications:
  - Machine translation
  - Summarization
  - Sentiment analysis
  - Chatbots and Virtual Assistants
  - Content Generation
  - etc…

Large Language Models

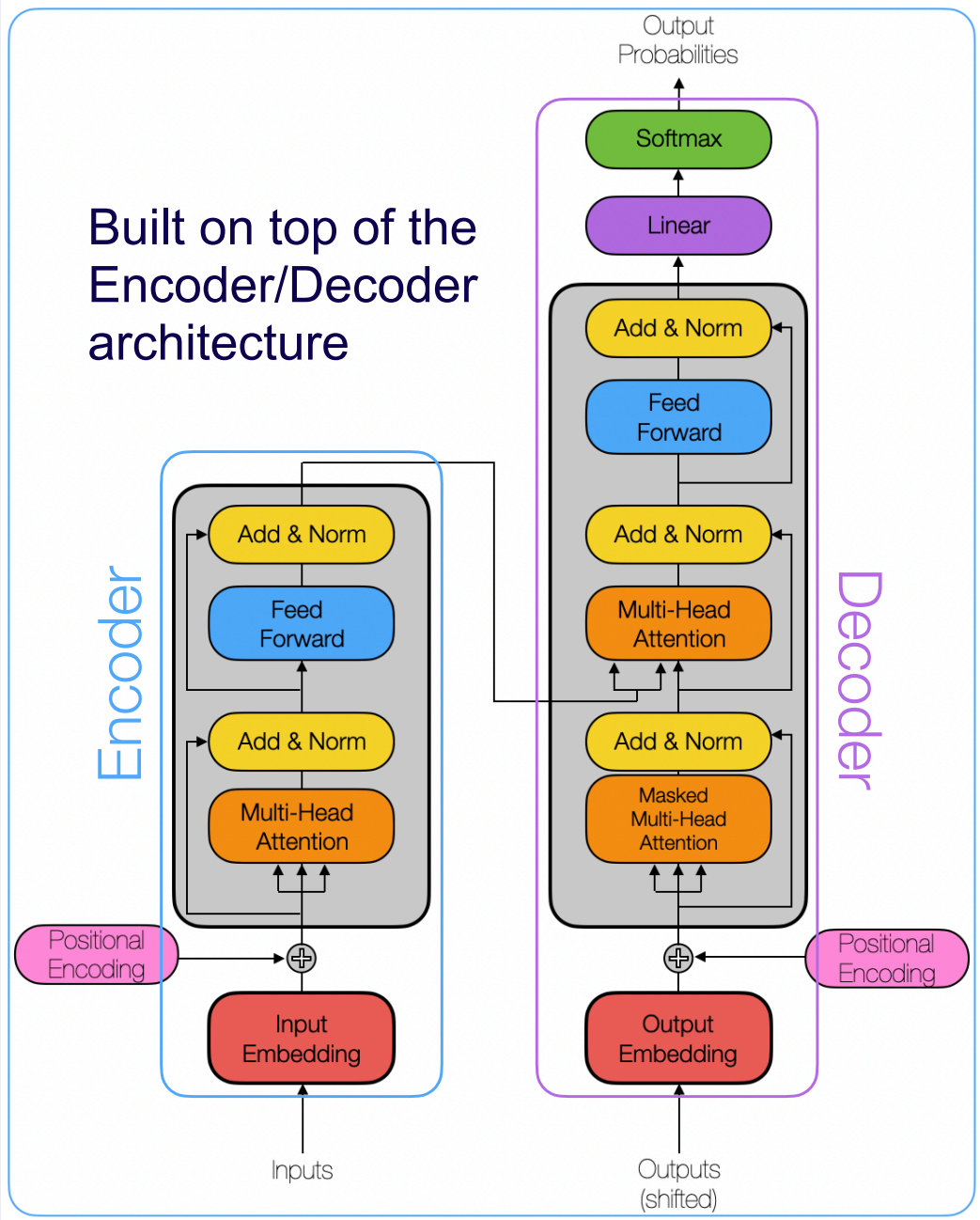
- Large Language Models (LLMs) are Deep Learning models trained on vast amounts of text data that are able to learn the complexities of language, including syntax, semantics, and context.
- Architecture:
  - Transformers: The backbone architecture of most LLMs, which uses self-attention mechanisms to process and generate text efficiently. Key examples include GPT (Generative Pre-trained Transformer), BERT (Bidirectional Encoder Representations from Transformers), and T5 (Text-to-Text Transfer Transformer).
  - Self-Attention: Allows the model to weigh the importance of different words in a sentence, making it capable of understanding context and relationships within the text.
  - Tokenization: Text is split into smaller units (tokens), which can be words, characters, or subwords. The model predicts the next token in a sequence based on previous ones.

Transformers



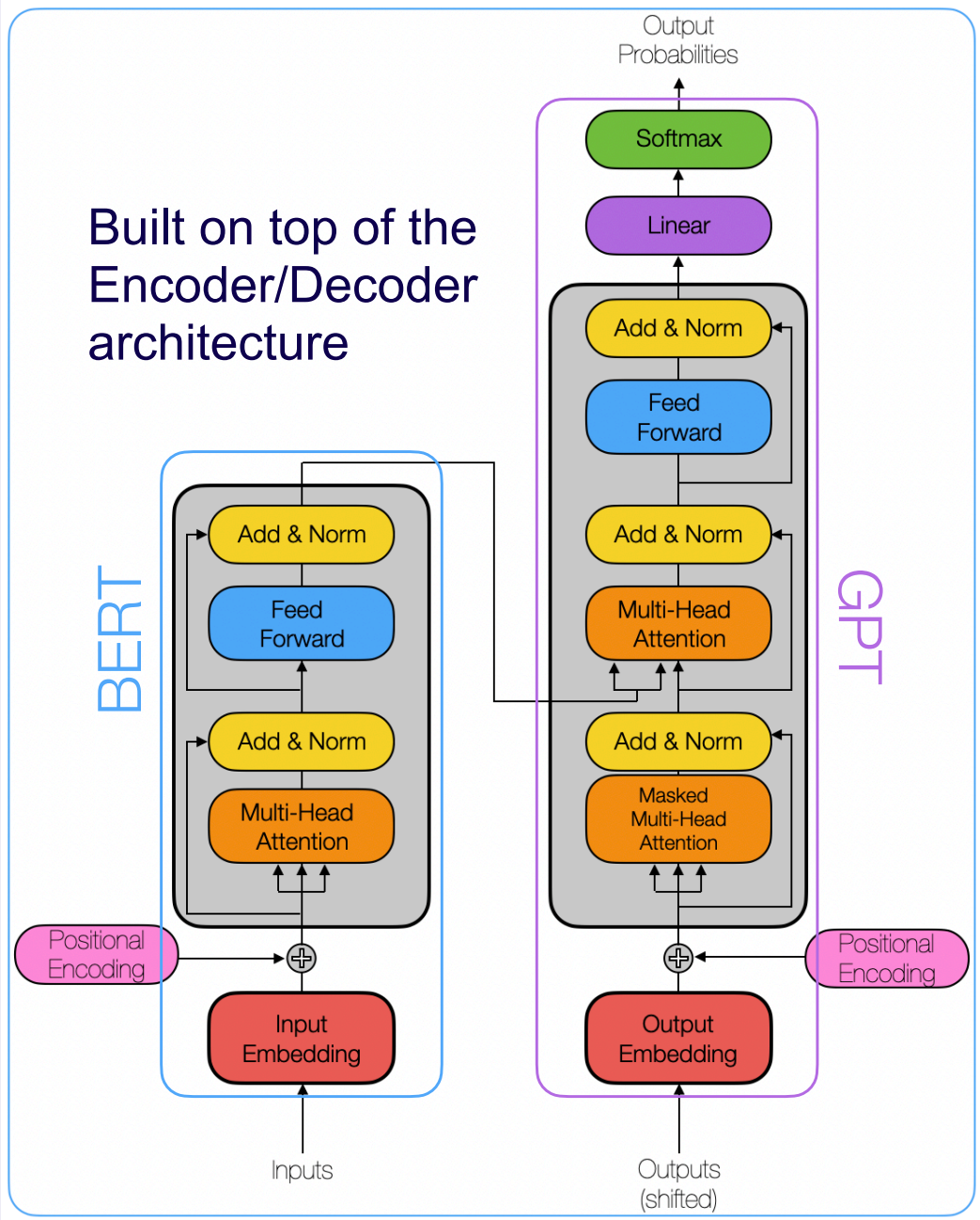
arXiv:1706.03762

Transformers



arXiv:1706.03762

Transformers

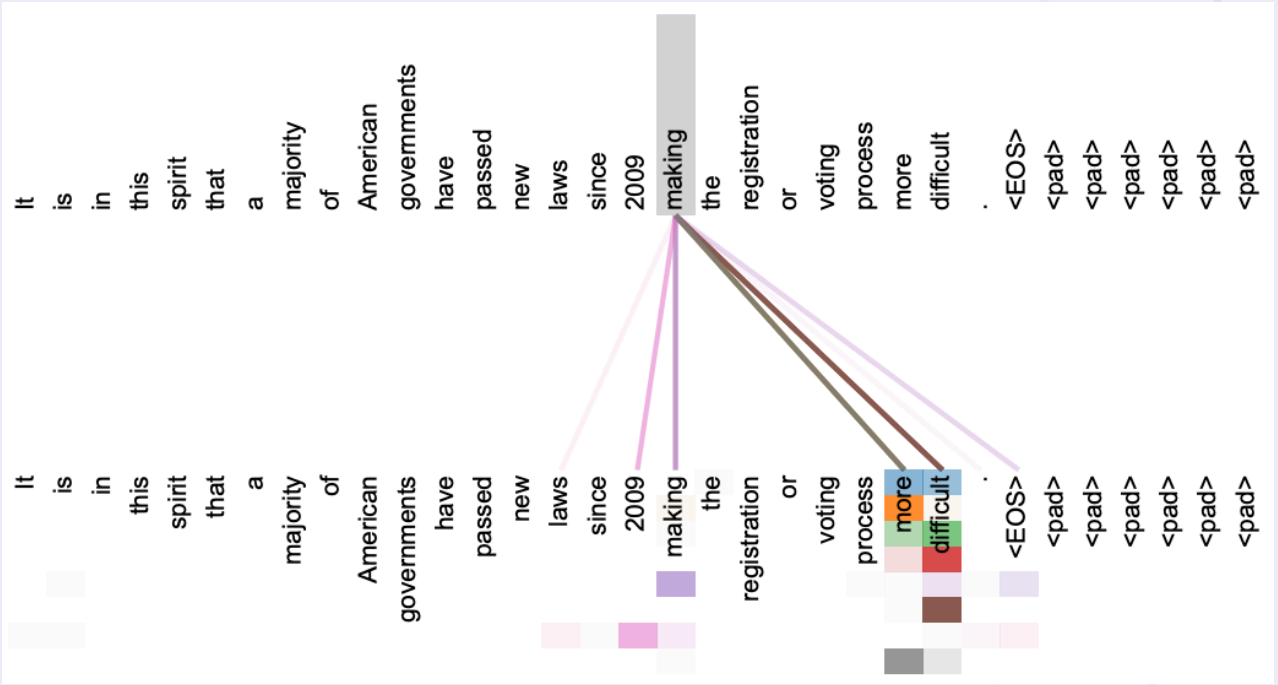


arXiv:1706.03762

Attention

arXiv:1706.03762

- “Simple” mechanism to allow each token to take the context it appears in into account
- Requires exponentially more weights to be computed (from each token to every other token)



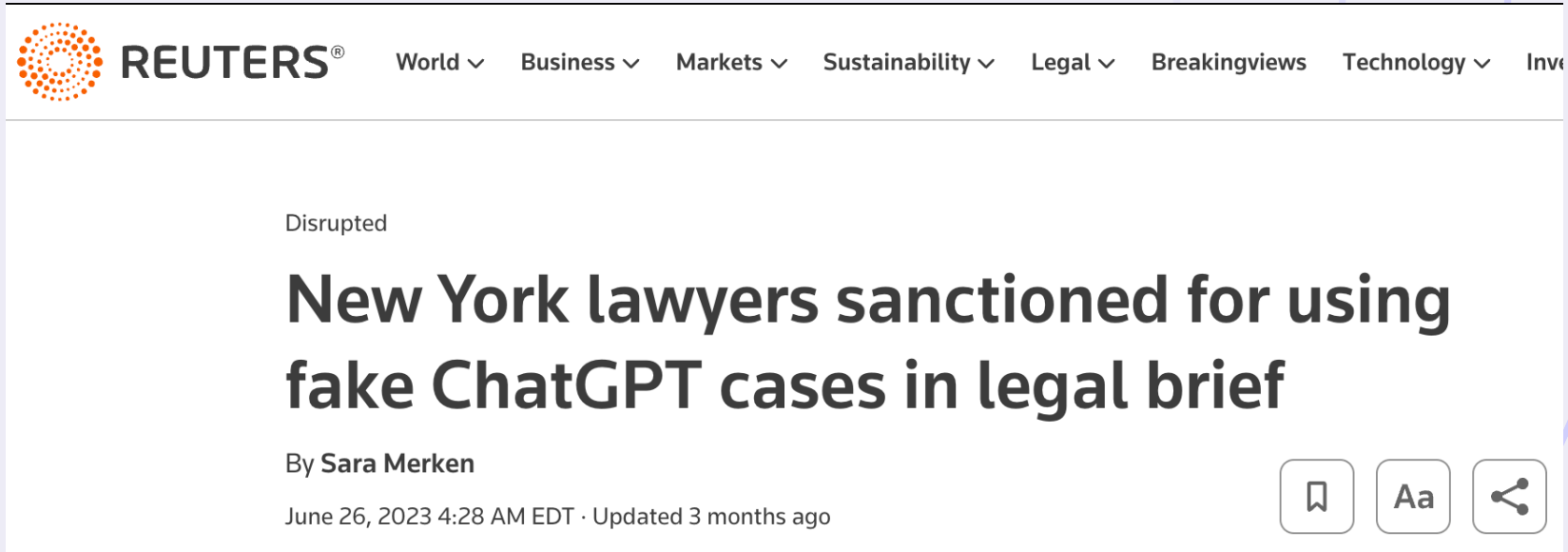
- Weights indicate the importance of each word relative to the current one

Hallucinations

- LLMs are just trying to guess the next word with limited or no understanding of what they're "talking about"
- The output produced can easily be non-sensical, or include information and details that are completely fabricated.

Hallucinations

- LLMs are just trying to guess the next word with limited or no understanding of what they're "talking about"
- The output produced can easily be non-sensical, or include information and details that are completely fabricated.
- A famous recent example:



https://www.reuters.com/legal/new-york-lawyers-sanctioned-using-fake-chatgpt-cases-legal-brief-2023-06-22/

Hallucinations

arXiv:2409.05746 (2024)

LLMs Will Always Hallucinate, and We Need to Live With This

Sourav Banerjee*
DataLabs
United We Care
sb@unitedwecare.com

Ayushi Agarwal
DataLabs
United We Care
ayushi@unitedwecare.com

Saloni Singla
DataLabs
United We Care
saloni@unitedwecare.com

September 10, 2024

Abstract

As Large Language Models become more ubiquitous across domains, it becomes important to examine their inherent limitations critically. This work argues that hallucinations in language models are not just occasional errors but an inevitable feature of these systems. We demonstrate that hallucinations stem from the fundamental mathematical and logical structure of LLMs. It is, therefore, impossible to eliminate them through architectural improvements, dataset enhancements, or fact-checking mechanisms. Our analysis draws on computational theory and Gödel's First Incompleteness Theorem, which references the undecidability of problems like the Halting, Emptiness, and Acceptance Problems. We demonstrate that every stage of the LLM process—from training data compilation to fact retrieval, intent classification, and text generation—will have a non-zero probability of producing hallucinations. This work introduces the concept of "Structural Hallucinations" as an intrinsic nature of these systems. By establishing the mathematical certainty of hallucinations, we challenge the prevailing notion that they can be fully mitigated.

Hallucinations

arXiv:2409.05746 (2024)

LLMs Will Always Hallucinate, and We Need to Live With This

Sourav Banerjee*
DataLabs
United We Care
sb@unitedwecare.com

Ayushi Agarwal
DataLabs
United We Care
ayushi@unitedwecare.com

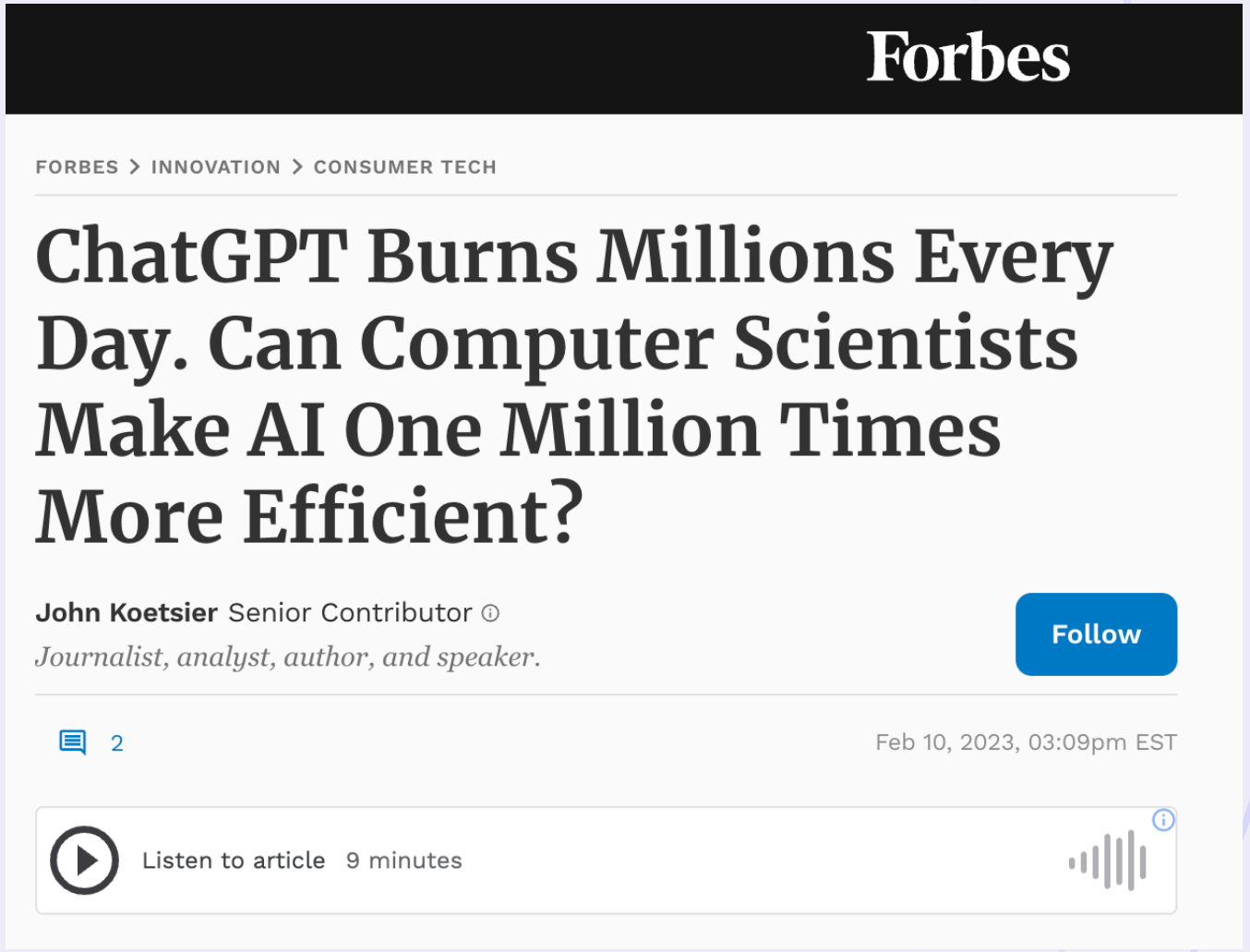
Saloni Singla
DataLabs
United We Care
saloni@unitedwecare.com

September 10, 2024

Abstract

As Large Language Models become more ubiquitous across domains, it becomes important to examine their inherent limitations critically. This work argues that hallucinations in language models are not just occasional errors but an inevitable feature of these systems. We demonstrate that hallucinations stem from the fundamental mathematical and logical structure of LLMs. It is, therefore, impossible to eliminate them through architectural improvements, dataset enhancements, or fact-checking mechanisms. Our analysis draws on computational theory and Gödel's First Incompleteness Theorem, which references the undecidability of problems like the Halting, Emptiness, and Acceptance Problems. We demonstrate that every stage of the LLM process—from training data compilation to fact retrieval, intent classification, and text generation—will have a non-zero probability of producing hallucinations. This work introduces the concept of "Structural Hallucinations" as an intrinsic nature of these systems. By establishing the mathematical certainty of hallucinations, we challenge the prevailing notion that they can be fully mitigated.

LLM infrastructure is expensive



Forbes

FORBES > INNOVATION > CONSUMER TECH

ChatGPT Burns Millions Every Day. Can Computer Scientists Make AI One Million Times More Efficient?

John Koetsier Senior Contributor

Journalist, analyst, author, and speaker.

Follow

2

Feb 10, 2023, 03:09pm EST

Listen to article 9 minutes

Pearson

LLM Use Cases in Data Science

- Data Preprocessing and Cleaning: data cleaning, transformation, parsing and normalization
- Data Exploration and Analysis: Feature engineering and data summarization
- Code Generation and Optimization: Write new code or refactor/optimize existing code.
- Natural Language to SQL: Convert plain language queries into SQL code and automate complex database interactions without deep SQL knowledge.
- Text Data Processing: Sentiment analysis, information extraction, topic modeling, and named entity recognition on unstructured text data.
- Knowledge Extraction from Scientific Papers: Summarize academic papers
- Data Augmentation and Synthesis: Generate synthetic data for model training, and create realistic scenarios for testing model robustness.

LLM Use Cases in Data Science

- Data Preprocessing and Cleaning: data cleaning, transformation, parsing and normalization
- Data Exploration and Analysis: Feature engineering and data summarization
- Code Generation and Optimization: Write new code or refactor/optimize existing code.
- Natural Language to SQL: Convert plain language queries into SQL code and automate complex database interactions without deep SQL knowledge.
- Text Data Processing: Sentiment analysis, information extraction, topic modeling, and named entity recognition on unstructured text data.
- Knowledge Extraction from Scientific Papers: Summarize academic papers
- Data Augmentation and Synthesis: Generate synthetic data for model training, and create realistic scenarios for testing model robustness.

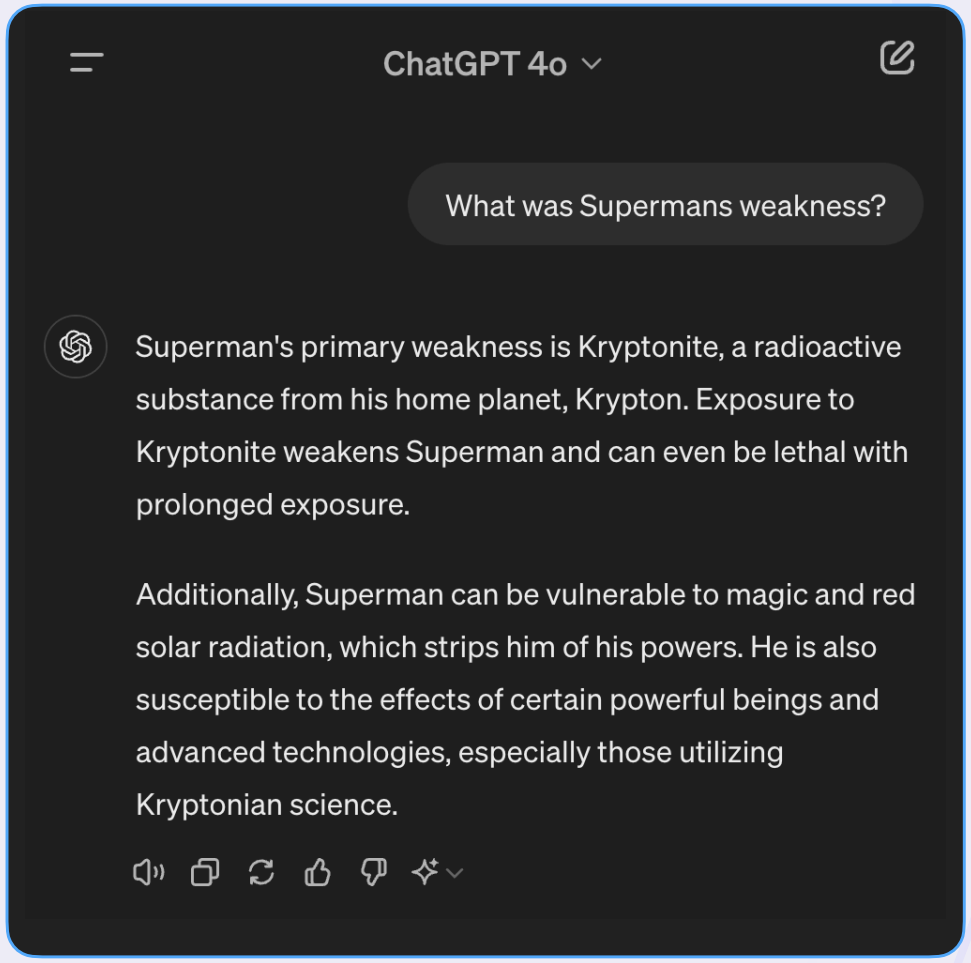
OpenAI

- American Research Lab, founded in 2015 by Ilya Sutskever, a former Google employee
- Heavily funded by Microsoft ($10B in 2023)
- Creator of some of the current state of the art models of Generative AI
- May 2020 - GPT-3
- Jan 2021 - DALL-E
- Aug 2021 - Codex
- Jul 2022 - DALL-E 2
- Nov 2022 - ChatGPT (based on GPT-3.5)
- Mar 2023 - GPT-4
- May 2024 - GPT-4o
- Sept 2024 - o1
- Dec 2024 - o3
- Feb 2025 - GPT-4.5
- Apr 2025 - GPT-4.1
- Aug 2025 - GPT-5
- Nov 2025 - GPT-5.1

Basic Usage

OpenAI

- You might be familiar with the basic web interface known as ChatGPT where you interact with the system through a basic text prompt as if you were text messaging your friends:



Pearson

Basic Usage

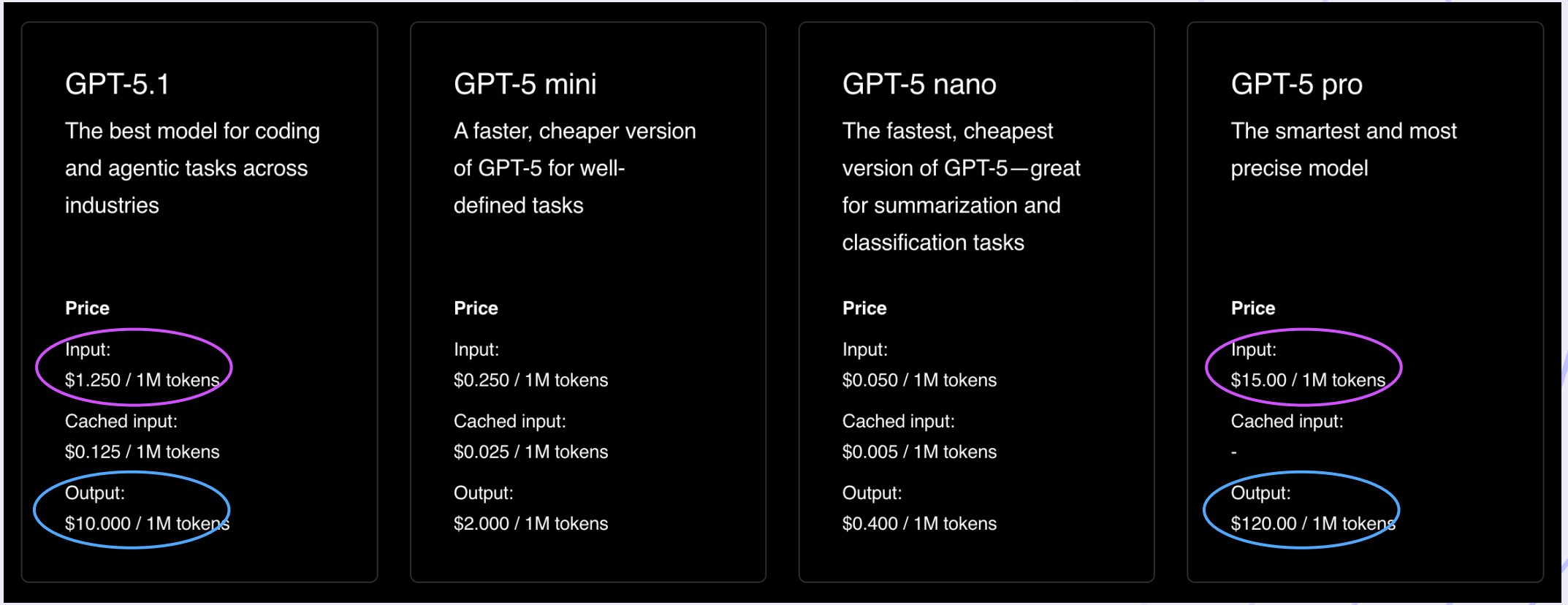
OpenAI

- Programmatically, things look a bit different
- The basic API call is chat.completions.create()
- It takes two required arguments:
  - model - The model to use. ChatGPT was originally based on gpt-3.5-turbo
  - messages - A list of dictionaries representing the conversation so far. Each element has several possible fields:
    - ”role" [required] - Three options
      - “system” - Instructs the model on how to behave
      - “user” - Represents user input
      - “assistant” - Corresponds to the output generated by the system
    - “content" [required] - Free-form text
    - ”name” [optional] - An optional name field to be used to identify the participants in the conversation

Pricing

OpenAI

- Unfortunately, OpenAI is not free to use, and the cost depends on the model used and the context size. Cost can vary by 12x from one model to another



GPT-5.1

The best model for coding and agentic tasks across industries

Price

Input:
$1.250 / 1M tokens

Cached input:
$0.125 / 1M tokens

Output:
$10.000 / 1M tokens

GPT-5 mini

A faster, cheaper version of GPT-5 for well-defined tasks

Price

Input:
$0.250 / 1M tokens

Cached input:
$0.025 / 1M tokens

Output:
$2.000 / 1M tokens

GPT-5 nano

The fastest, cheapest version of GPT-5—great for summarization and classification tasks

Price

Input:
$0.050 / 1M tokens

Cached input:
$0.005 / 1M tokens

Output:
$0.400 / 1M tokens

GPT-5 pro

The smartest and most precise model

Price

Input:
$15.00 / 1M tokens

Cached input:
-

Output:
$120.00 / 1M tokens

https://openai.com/api/pricing/

Pearson

Rate Limits

OpenAI

- Rate limits are imposed to prevent DDoS attacks and misconfigured applications from running rampant

MODEL	TOKEN LIMITS	REQUEST AND OTHER LIMITS	BATCH QUEUE LIMITS
gpt-4o	450,000 TPM	5,000 RPM	1,350,000 TPD
gpt-3.5-turbo	80,000 TPM	5,000 RPM	400,000 TPD
gpt-4	40,000 TPM	5,000 RPM	200,000 TPD
gpt-4-turbo	450,000 TPM	500 RPM	1,350,000 TPD
text-embedding-3-small	1,000,000 TPM	5,000 RPM	20,000,000 TPD
dall-e-3		7 images per minute	
tts-1		50 RPM	
whisper-1		50 RPM	

https://openai.com/api/pricing/

Hugging Face

- Founded in 2016 by French entrepreneurs in New York City
- Originally a chatbot aimed at teenagers, eventually pivoted to being a Machine Learning platform that allows users to share code and datasets for NLP based models, specially LLMs
- Creator of the transformers package and the BLOOM LLM

Basic Usage

Hugging Face

- The transformers package implements the pipeline() method
- Pipelines are objects that abstract away most of the complexity of interacting with each individual model behind a simple and consistent API

Basic Usage

Hugging Face

- The basic API call is pipeline() that takes two main arguments and returns a Pipeline object:
  - task - A string identifying the task we are trying to perform. Common examples:
    - ‘fill-mask' - Identify possible values for the masked word
    - ‘ner' - Named Entity Recognition
    - ‘question-answering‘ - Answer questions about the input
    - ‘translation_X_to_Y' - Translate from language X to language Y
    - ‘text-generation‘ - Generate text based on a prompt
  - model [Optional] - the name of the model to use. Each pipeline has a reasonable default value
  - Each task can take further relevant arguments

🦜🔗 LangChain

- Launched in 2022 as an open source project by Harrison Chase
- Incorporated as a company in Jan 2023
- LangChain is a Swiss army knife of wrappers for a huge range of LLMs, database and storage solutions, data formats, etc.
- LangChain revolves around the concept of a pipeline (similar in spirit to the HuggingFace ones) that can be implemented using a wide range of modules to manage and process data from initial input all the way to final output

🦜🔗 LangChain

- LangChain is an open-source Python framework designed to simplify the development of applications using large language models (LLMs).
- Main features:
  - Model Abstraction: LangChain manages inputs and outputs seamlessly by abstracting away model details
  - Modular Structure: Allows developers to combine different prompts and even multiple
- LLMs within a single application.
  - Chains: Multiple components can be chained together to create complex applications
  - Data Integration: Allows LLMs to transform, store, and retrieve data from databases and online sources.
  - Memory Management: Provides utilities for adding memory to LLM systems, retaining conversation history or summaries for context.

🦜🔗 LangChain

- LangChain can be used to build a wide range of LLM-powered applications:
  - Chatbots
  - Coding assistants and code security analysis
  - Recommendation systems
  - Document analysis and summarization
  - Question-answering systems
  - Data augmentation
  - Text classification and summarization
  - Sentiment analysis
  - Machine translation

🦜🔗 LangChain

- A Chain is a sequence of calls to components, including other chains.
- Conceptually similar to sklearn Pipelines
- Component Examples:
  - LLMMath - For math related queries
  - SQLDatabaseChain - To query databases
  - OpenAIModerationChain - Check content against moderation rules
  - ConstitutionalChain - Legal applications
  - LLMCheckerChain - Prevent hallucinations

Generative AI for Data Science

https://github.com/DataForScience/LLM4DS

Pearson

2. Prompt Engineering for Data Science

Output Formatting

- We can obtain structured output from LLMs by guiding the model to produce data in formats like JSON, XML, or CSV.
- Different Approaches:
  - Explicit Instructions: Clearly specify the desired format in your prompt.
  - Use Templates: Include a sample of the expected output structure
- Post-processing:
  - Parsing Libraries: Use JSON, XML, etc parsers to validate the output.
  - Error Correction Scripts: Automatically detect and correct common formatting errors.

Prompting

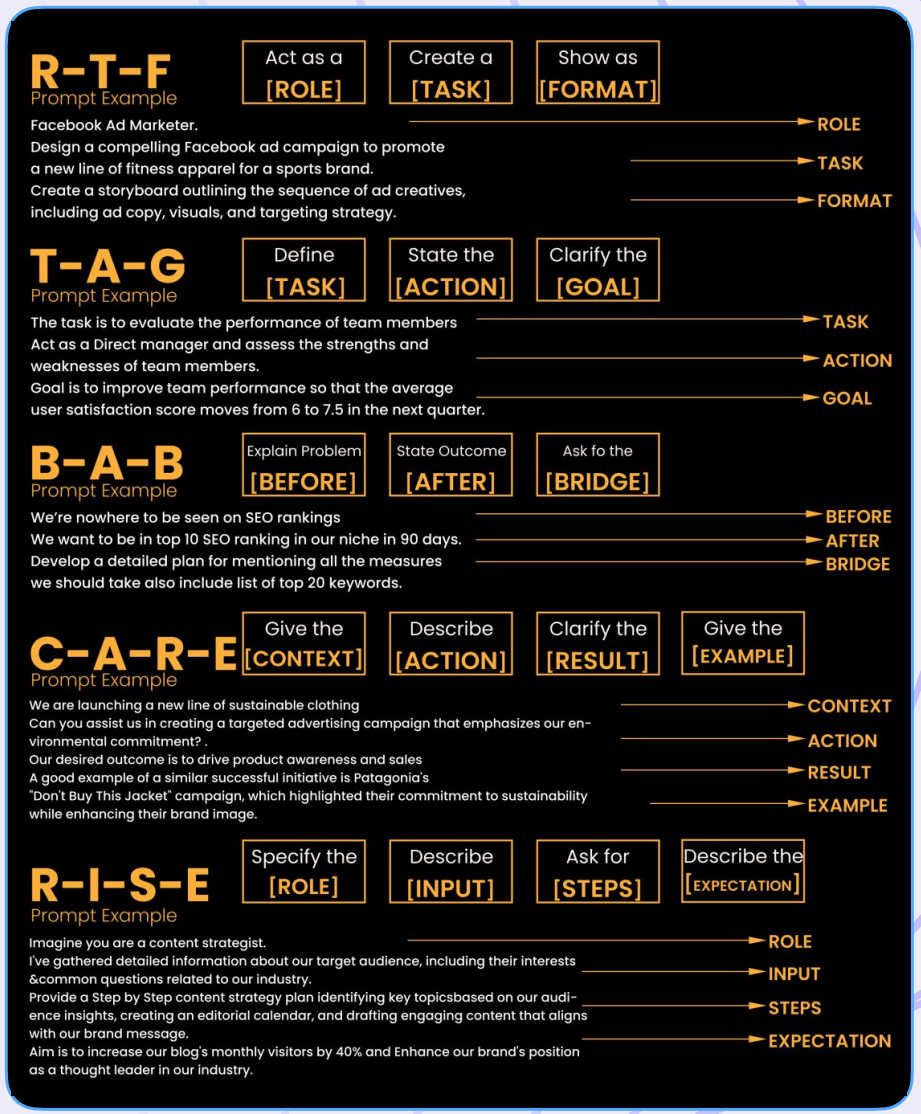
- Prompting is how we interact with LLMs
- Choosing the correct type of prompt can have a major effect on the results we obtain
- Common prompting approaches:
  - Zero-Shot - Explain the task and ask for the result
  - Few-Shot - Provide a few examples of both inputs and outputs that the model can generalize from
  - Chain of Thought - Ask the model to explain how it reached the result

Prompting Tips

- Start simple and integrate by gradually adding the details necessary to improve results
- Break down complex tasks into smaller, more manageable, subtasks
- Use action commands ("Write", "Classify", "Summarize", "Translate", "Order", etc.) to indicate what the goal is
- More descriptive and detailed the prompts produce better results.

Structured Prompts

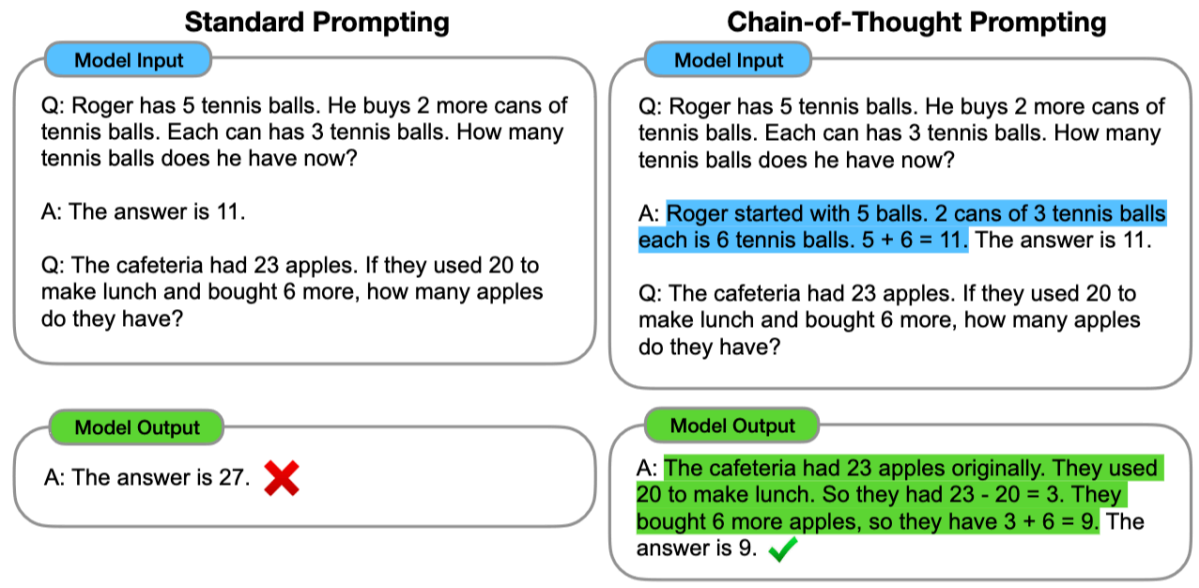
- Structured Prompts can significantly improve the quality of the output. They:
  - Help the model concentrate on relevant information, improving the quality of responses.
  - Ensures the model understands the expected perspective and objective.
  - Reducing ambiguity in the model's response.
  - Reduce the likelihood of formatting errors



Chain-of-Thought prompts

arXiv: 2201.11903 (2022)

- Chain of Thought (CoT) prompts encourages LLMs to generate intermediate steps before providing the final solution to a problem.
- CoT breaks down complex problems into manageable, intermediate steps and mimic an human thought process when working through multi-step problems.
- Relatively small change to the prompt can have a profound effect

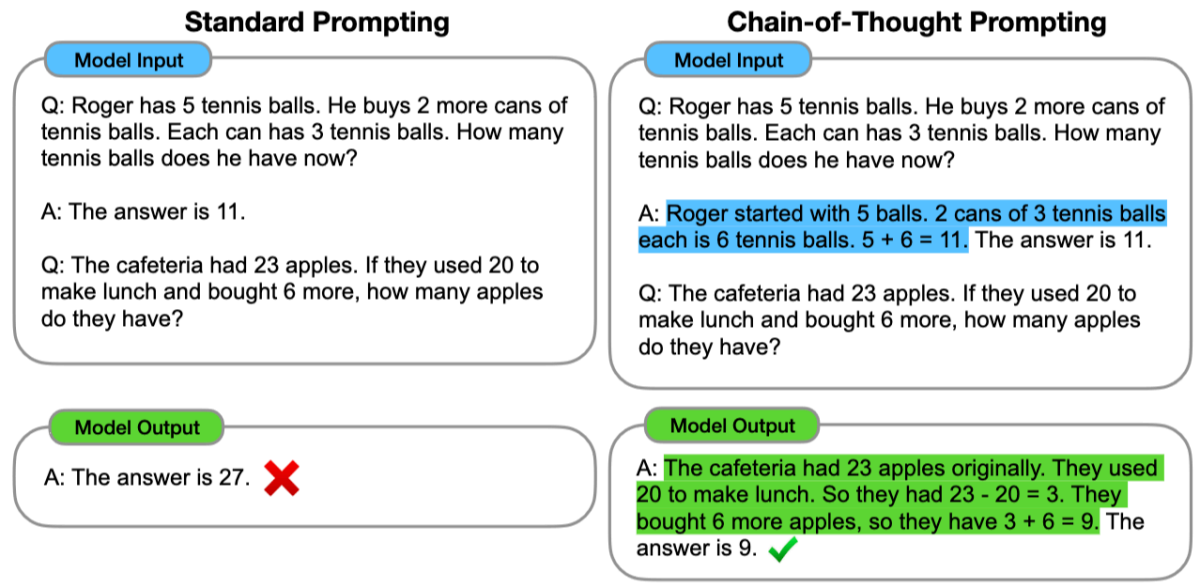


Pearson

Chain-of-Thought prompts

arXiv: 2201.11903 (2022)

- Chain of Thought (CoT) prompts encourages LLMs to generate intermediate steps before providing the final solution to a problem.
- CoT breaks down complex problems into manageable, intermediate steps and mimic an human thought process when working through multi-step problems.
- Relatively small change to the prompt can have a profound effect

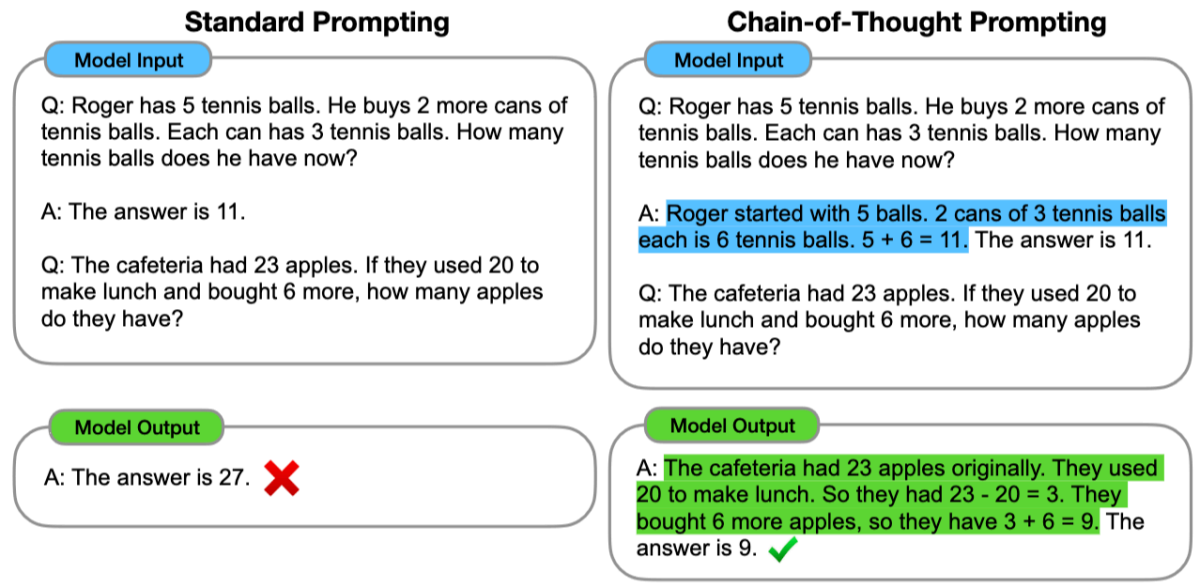


Add an example to break down the problem

Chain-of-Thought prompts

arXiv: 2201.11903 (2022)

- Chain of Thought (CoT) prompts encourages LLMs to generate intermediate steps before providing the final solution to a problem.
- CoT breaks down complex problems into manageable, intermediate steps and mimic an human thought process when working through multi-step problems.
- Relatively small change to the prompt can have a profound effect

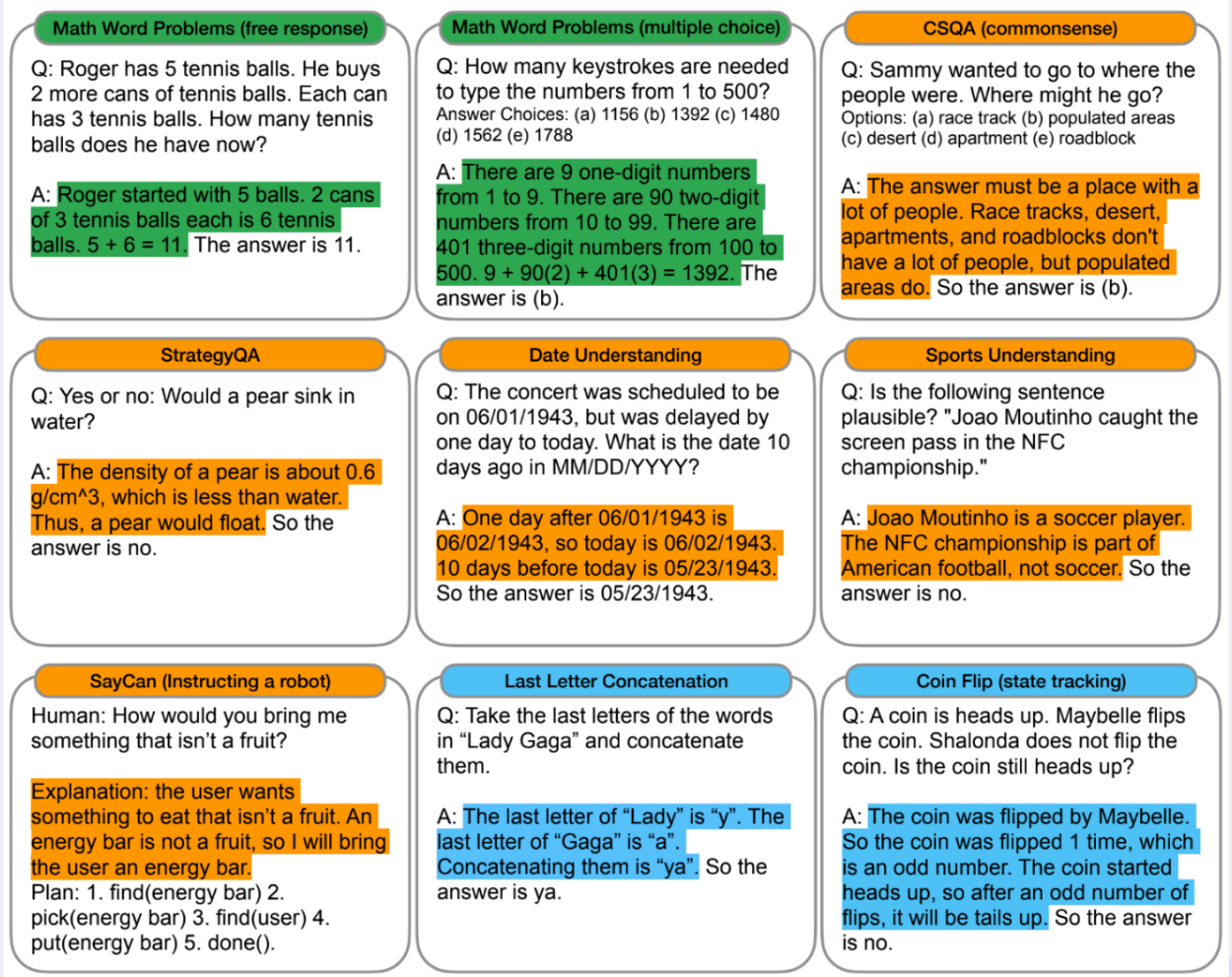


Add an example to break down the problem

LLM follows similar detailed approach to reach a better solution

Chain-of-Thought prompts

arXiv: 2201.11903 (2022)



Math Word Problems (free response)

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Math Word Problems (multiple choice)

Q: How many keystrokes are needed to type the numbers from 1 to 500?
Answer Choices: (a) 1156 (b) 1392 (c) 1480 (d) 1562 (e) 1788

A: There are 9 one-digit numbers from 1 to 9. There are 90 two-digit numbers from 10 to 99. There are 401 three-digit numbers from 100 to 500. 9 + 90(2) + 401(3) = 1392. The answer is (b).

CSQA (commonsense)

Q: Sammy wanted to go to where the people were. Where might he go?
Options: (a) race track (b) populated areas (c) desert (d) apartment (e) roadblock

A: The answer must be a place with a lot of people. Race tracks, desert, apartments, and roadblocks don't have a lot of people, but populated areas do. So the answer is (b).

StrategyQA

Q: Yes or no: Would a pear sink in water?

A: The density of a pear is about 0.6 g/cm^3, which is less than water. Thus, a pear would float. So the answer is no.

Date Understanding

Q: The concert was scheduled to be on 06/01/1943, but was delayed by one day to today. What is the date 10 days ago in MM/DD/YYYY?

A: One day after 06/01/1943 is 06/02/1943, so today is 06/02/1943. 10 days before today is 05/23/1943. So the answer is 05/23/1943.

Sports Understanding

Q: Is the following sentence plausible? "Joao Moutinho caught the screen pass in the NFC championship."

A: Joao Moutinho is a soccer player. The NFC championship is part of American football, not soccer. So the answer is no.

SayCan (Instructing a robot)

Human: How would you bring me something that isn't a fruit?

Explanation: the user wants something to eat that isn't a fruit. An energy bar is not a fruit, so I will bring the user an energy bar.
Plan: 1. find(energy bar) 2. pick(energy bar) 3. find(user) 4. put(energy bar) 5. done().

Last Letter Concatenation

Q: Take the last letters of the words in "Lady Gaga" and concatenate them.

A: The last letter of "Lady" is "y". The last letter of "Gaga" is "a". Concatenating them is "ya". So the answer is ya.

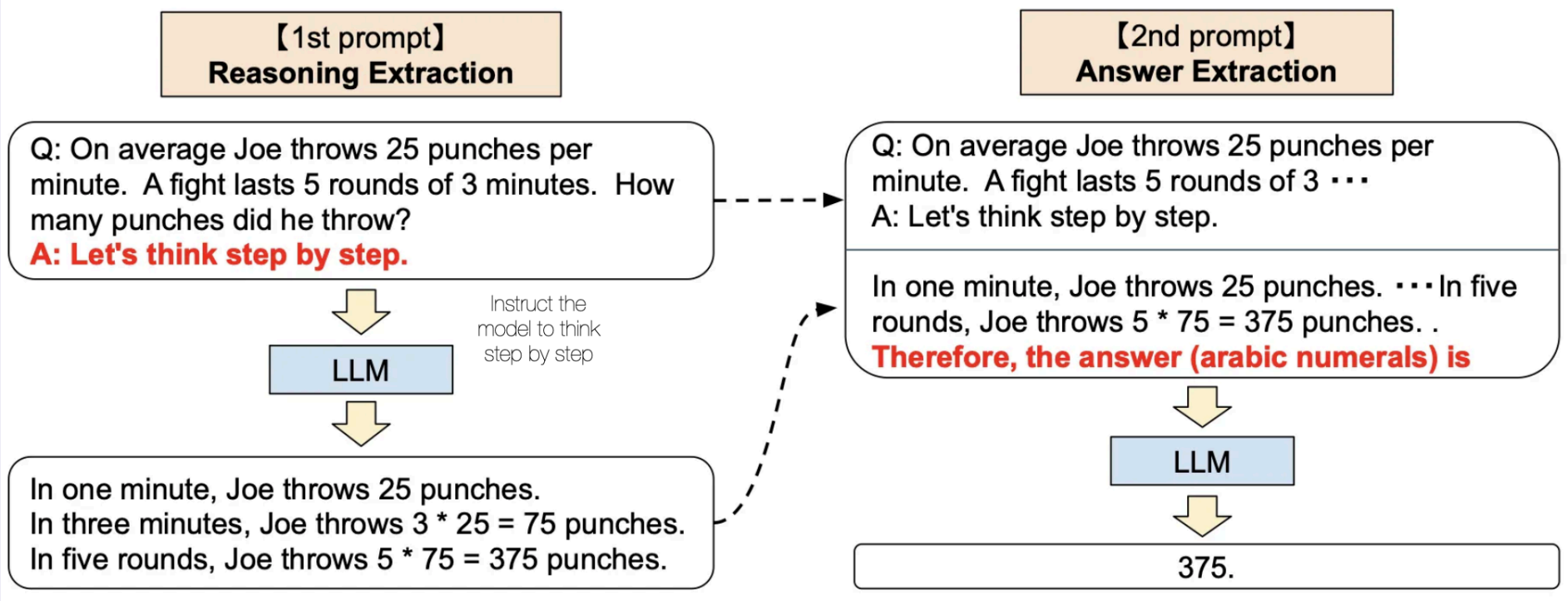
Coin Flip (state tracking)

Q: A coin is heads up. Maybelle flips the coin. Shalonda does not flip the coin. Is the coin still heads up?

A: The coin was flipped by Maybelle. So the coin was flipped 1 time, which is an odd number. The coin started heads up, so after an odd number of flips, it will be tails up. So the answer is no.

Chain-of-Thought - Zero Shot

arXiv: 2201.11903 (2022)



Pearson

Prompt Engineering

https://github.com/DataForScience/LLM4DS

Pearson

3. NLP with HuggingFace

Advantages of LLMs for NLP Tasks

- LLMs excel in various traditional NLP tasks by leveraging their ability to understand context, nuance, and syntax in text
- Advantages:
  - Versatility - The same model can handle a large number of tasks
  - Minimal or no task specific training - Using One-Shot or Few-Shot prompting techniques
  - Ease of Integration - Tools like the OpenAI, Hugging Face, or LangChain facilitate the integration of LLMs into new or pre-existing data processing pipelines

Disadvantages of LLMs for NLP Tasks

- LLMs excel in various traditional NLP tasks by leveraging their ability to understand context, nuance, and syntax in text
- Caveats:
  - Computationally Intensive - Training and Inference require significant computational resources
  - Bias and Fairness - LLMs reflect any biases that were present in their training data
  - Hallucinations - LLMs can generate plausible-sounding outputs that are completely incorrect.

Named Entity Recognition

- Named Entity Recognition (NER) is a traditional NLP technique used to classify key entities within text into predefined categories. Entities cam be names of people, organizations, locations, dates, numerical expressions, and other domain-specific terms.
- Applications:
  - Information Extraction: Extract key facts from documents
  - Search and Information Retrieval: Help search engines provide more precise search results by indexing and recognizing named entities
  - Customer Support Automation
  - Document Summarization.
  - Data Enrichment: Connect entities to external data sources
- LLMs are a significant improvement over previous approaches due to their ability to leverage context to best identify entities

Part-of-Speech Tagging

- Part of Speech (POS) Tagging labels words in a snippet of text with their corresponding part of speech (e.g., Noun, Verb, Adjective) to better understand the syntactic structure of sentences, which is crucial for downstream tasks like parsing, named entity recognition, and text generation.
- Applications:
  - Syntax Analysis: Understand the grammatical structure of sentences
  - Disambiguation: Remove ambiguities in words whose meaning is context dependent
  - Feature Extraction: Generate linguistic features for others tasks like NER and sentiment analysis.
- LLMs use attention mechanisms to capture complex dependencies and contextual information

Summarization

- Summarization is the process of generating a shorter version of a longer piece of text while preserving its main ideas and essential information.
- Two main approaches to Summarization:
  - Extractive - Selects key sentences, phrases, or segments directly from the original text to identify and extract the most informative parts of the text.
  - Abstractive - Generates a summary by generating novel sentences that capture the essence of the document.
- LLMs generally produce abstractive summarization but can also be fine-tuned to produce extractive summarization.

Question Answering

- Question Answering (QA) techniques aim to provide answers to user queries based on a given context or knowledge base.
- Many points in common to Summarization
- Two main approaches of QA:
  - Extractive - Selects key sentences, phrases, or segments directly from the original text that contain the answer.
  - Abstractive - Generates an answer by generating novel sentences that answer the query based on the context provided
- Two main classes of QA:
  - Closed-Domain - Limited to well defined domains
  - Open-Domain - No predefined constraints

Sentiment Analysis

- Sentiment Analysis is a traditional area in Natural Language Processing (NLP) that aims to determine the emotional tone or attitude expressed in text.
- Traditional methods are either lexicon-based (counting positive/negative words from a dictionary) or Machine Learning based (like Naive Bayes or SVM).
- Applications:
  - Opinion Mining extracting actionable insights from large volumes of user-generated content.
  - Brand Monitoring
  - Customer Service

Sentiment Analysis

- LLM advantages:
  - Nuance: Accurately classify text even when it involves sarcasm, irony, or complex sentence structures, which often defeats traditional models.
  - Zero- and Few-Shot Learning making deployment much faster.
  - Aspect-Based Analysis: Go beyond overall sentiment to identify the sentiment toward specific aspects within a text (e.g., "The phone's camera is great, but the battery life is terrible").

NLP with HuggingFace

https://github.com/DataForScience/LLM4DS

Pearson

4. Text to Speech with OpenAI

Speech to text

- Two speech to text tasks:
  - Transcriptions
  - Translations
- Originally, both tasks were supported by the Whisper model. Currently transcriptions is also available with:
  - gpt-4o-mini-transcribe
  - gpt-4o-transcribe
  - gpt-4o-transcribe-diarize

The Whisper model

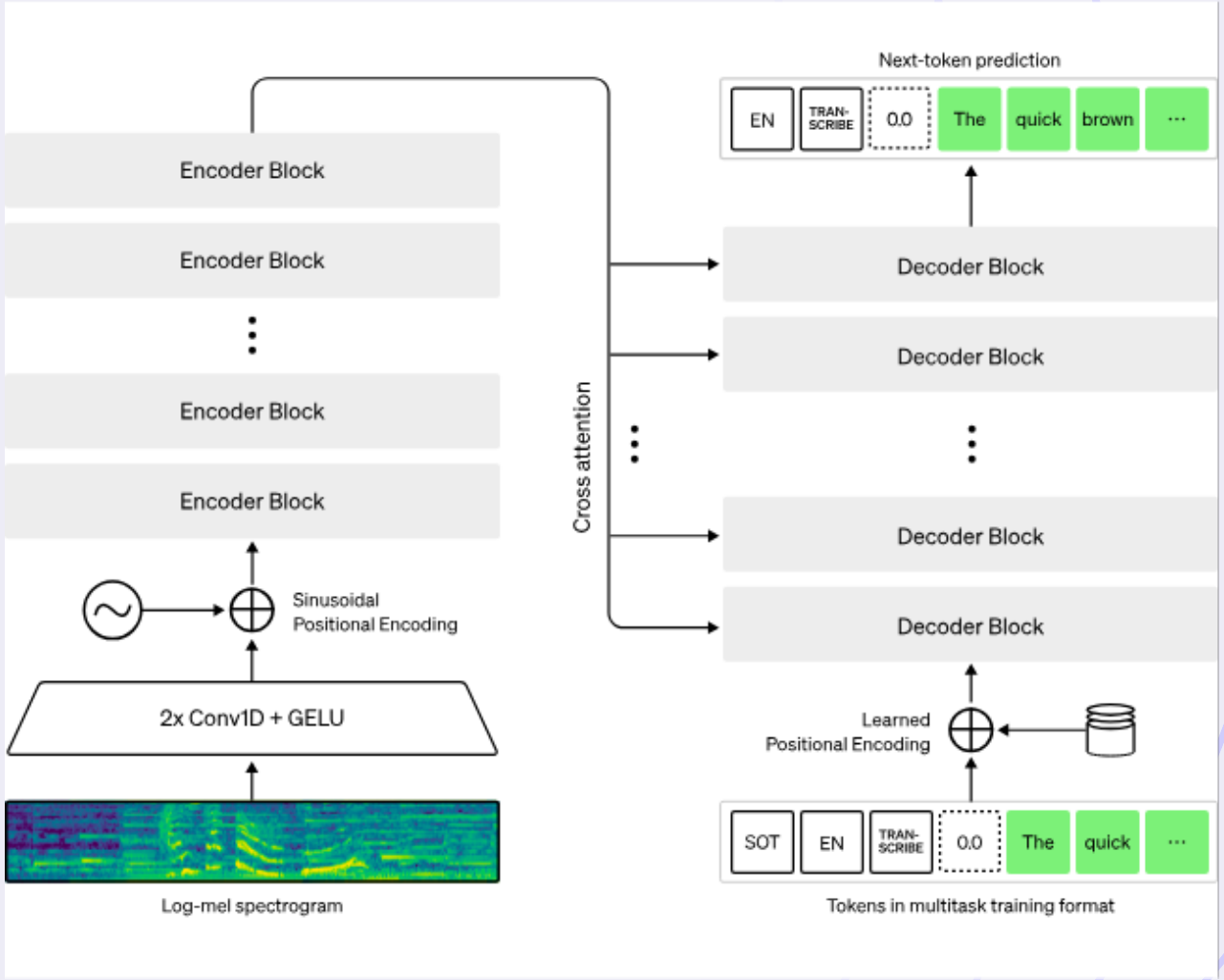
arXiv: 2212.04356 (2022)

- Whisper is an Automatic Speech Recognition (ASR) system trained on multilingual and multitask supervised data collected from the web.
- Originally introduced in Sep. 2022, the most recent version is from Nov. 2023
- Features
  - Text to Speech
  - Audio Transcription
  - Automatic Translation

Whisper Architecture

arXiv: 2212.04356 (2022)

- Basically a modified transformer!



Pearson

Audio transcription

- The basic API call is audio.transcriptions.create()
- It takes several required arguments:
  - model - The model to use, gpt-4o-transcribe
  - file - The path to the audio file to use,
  - response_format - What format to return the transcription, “json”, “text”

Text to Speech

- The main text to speech task is to produce spoken audio based on written text
- Current state of the art model is gpt-4o-mini-tts: but tts-1 and tts-1-hd still remain available.

- Output formats:
  - MP3: The default
  - Opus: For low latency streaming.
  - AAC: Preferred by YouTube, Android, iOS.
  - FLAC: Lossless audio compression.
  - WAV: Uncompressed audio

Text to Speech with OpenAI

https://github.com/DataForScience/LLM4DS

Pearson

Contacts

Bruno Gonçalves

https://data4sci.com

info@data4sci.com

https://data4sci.com/call

https://www.linkedin.com/in/bmtgoncalves/

Bruno Gonçalves is an author, public speaker, corporate trainer, and consultant specializing in Generative AI, Blockchain Analytics, and Machine Learning. He founded Data For Science, Inc in 2009 to help individuals and companies solve their data driven problems.