Jason Tran | Michelle Duong | AaJanae Henry

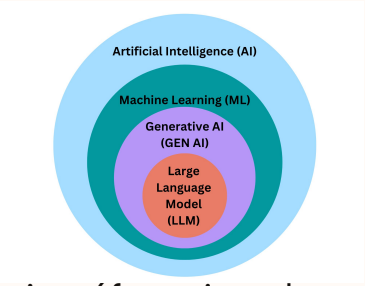
Am I (AI) Biased

Overview

Focused on building foundational knowledge of LLMs and hands-on experimentation. Loaded and ran text generation with 3 models:

- Llama-3.2-1B – a smaller, faster model for basic text generation tasks.
- Llama-3.2-3B – a more capable model, demonstrating how scaling up increases fluency and context understanding.
- Phi-3.5-Mini-Instruct – an instruction-tuned model that follows prompts more precisely and provides structured responses.

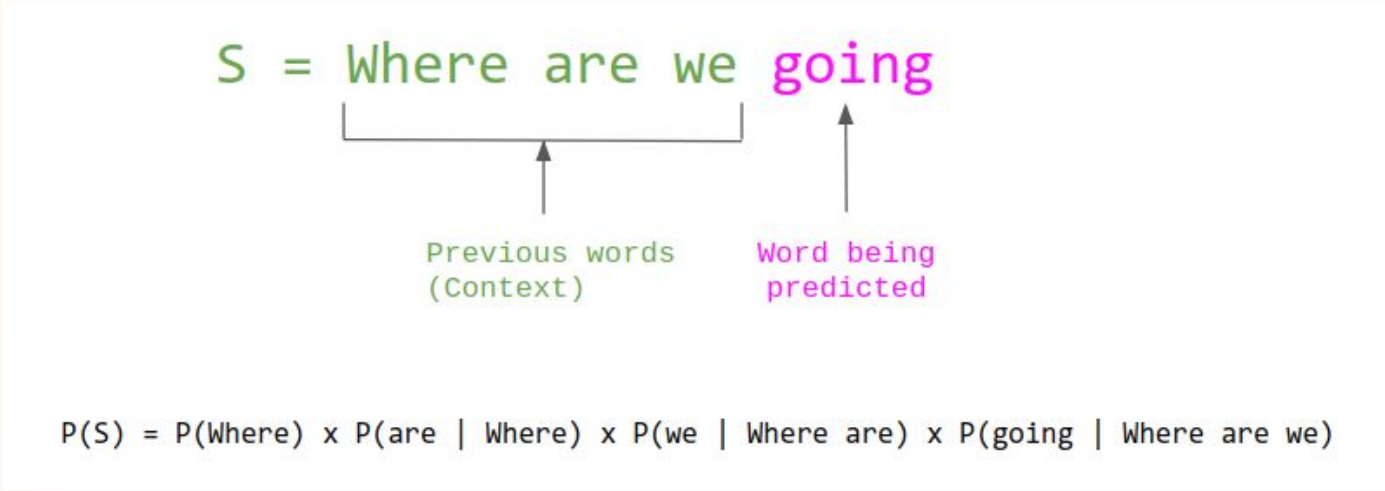
Language Models



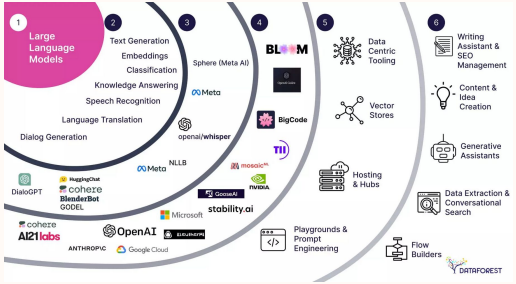
- A **language model** is a probability distribution (function that gives the probabilities of occurrence of possible events) over sequences of tokens (words)
  - *p(W) = p(x1, x2, x3, … , xL)*, where
    - W represents a sequence of words
    - x1, x2, x3, … , xL represents individual words
    - **Example**: p(hi, my, name, is, connor)
  - *p(xL | x1, x2, x3, … , xL-1)*
    - Probability of an upcoming word
    - **Example**: p(next word | current word)
- Assigns each sequence of tokens a probability
  - i.e., A probability to every possible sentence
- Higher Probability = more “correct” or probable sentence
- Used to predict and generate human like texts

Language Models Examples

- p(the,mouse,ate,the,cheese) = 0.02
- p(the,cheese,ate,the,mouse) = 0.01
- p(mouse,the,the,cheese,ate) = 0.0001

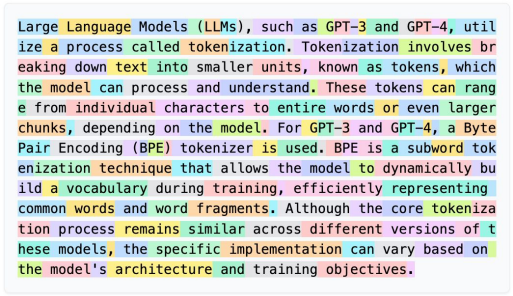


Large Language Models

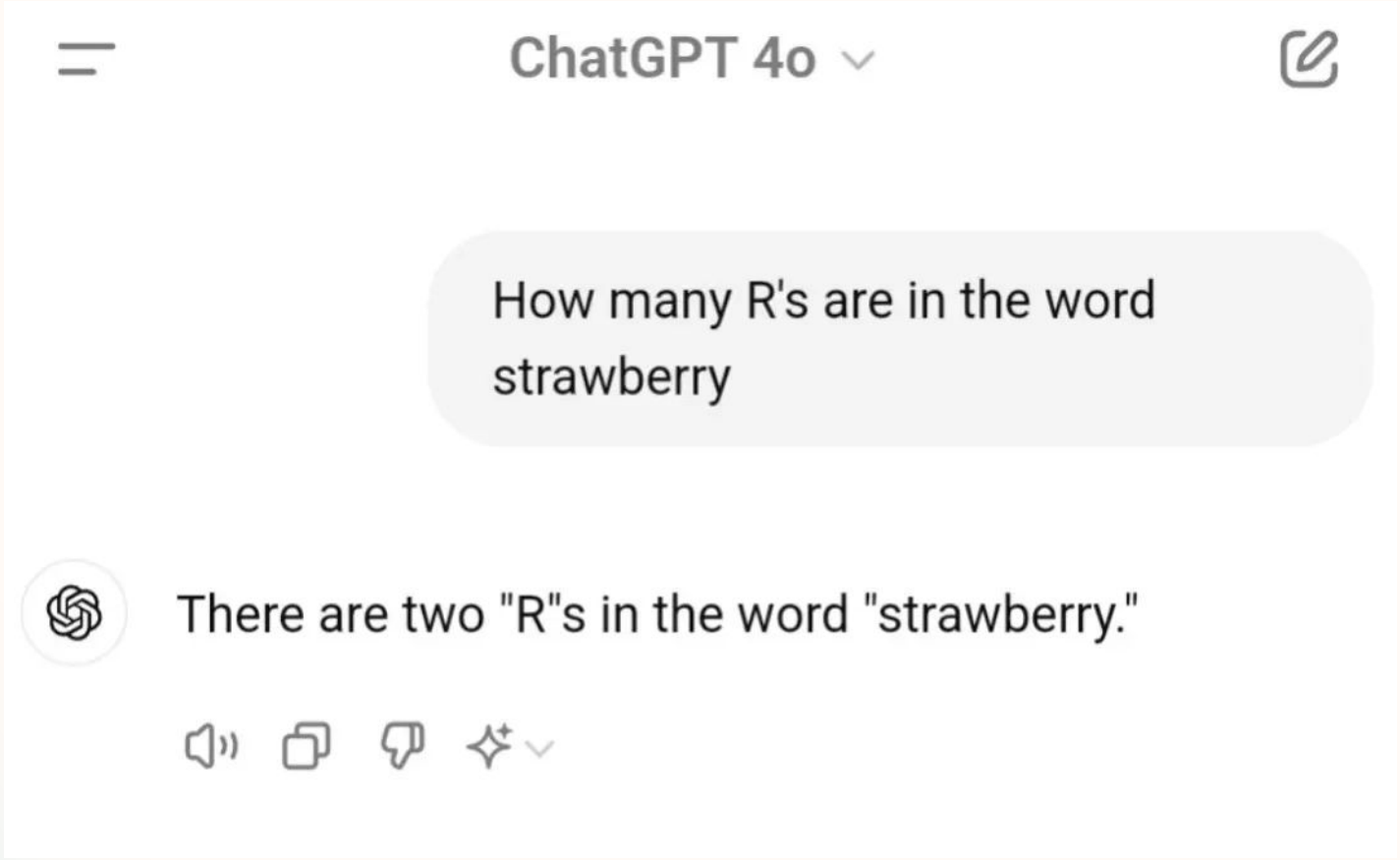


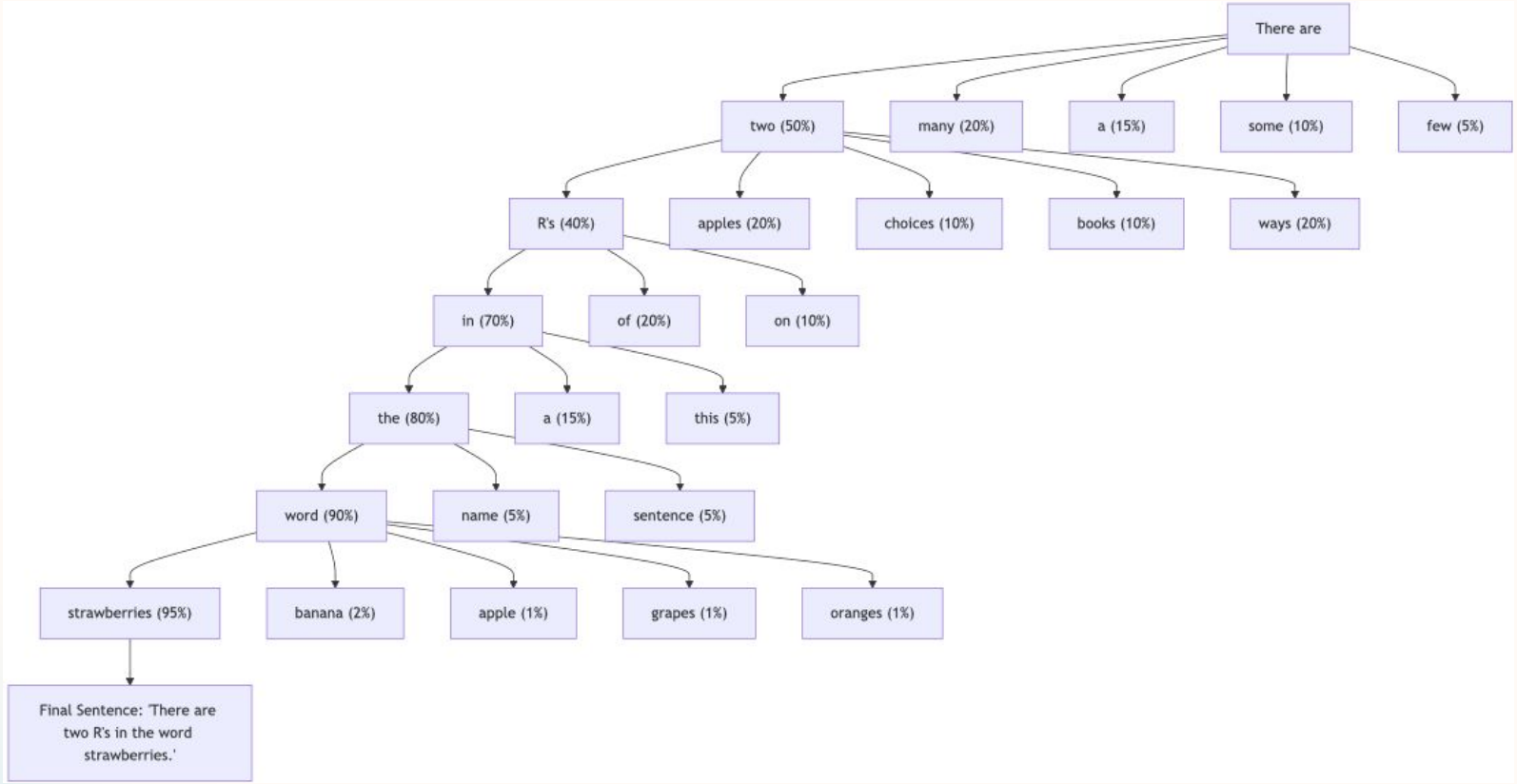
- A **large language model** (LLM) is a language model trained with vast amounts of data to understand, generate, and accurately predict human language
  - This training involves a massive amount of data, with GPT-4 using ~13 trillion tokens
- Uses machine learning to identify patterns and relationships in language.
- Trained with self-supervised learning for natural language processing tasks, especially language generation.

Tokenization



- Recall how a language model is a probability distribution over a sequence of tokens
- But natural language text appears as a string
  - “Hi, my name is Connor”
- A **tokenizer** converts any string into a sequence of tokens that an LLM can understand and process
  - Word-Based Tokenization
    - [“Hi”, “,”, “my”, “name”, “is”, “Connor”]
  - Character-Based Tokenization
    - [“H”, “i”, “,”, “m”, “y”, “n”, “a”, “m”, “e”, “i”, “s”, “C”, “o”, “n”, “n”, “o”, “r”]
  - Subword-Based Tokenization
    - [“hi”, “,”, “my”, “name”, “is”, “Conn”, “or”]


ChatGPT 4o
How many R's are in the word strawberry
There are two "R"s in the word "strawberry."


There are
two (50%)
many (20%)
a (15%)
some (10%)
few (5%)
R's (40%)
apples (20%)
choices (10%)
books (10%)
ways (20%)
in (70%)
of (20%)
on (10%)
the (80%)
a (15%)
this (5%)
word (90%)
name (5%)
sentence (5%)
strawberries (95%)
banana (2%)
apple (1%)
grapes (1%)
oranges (1%)
Final Sentence: 'There are two R's in the word strawberries.'

Tokenization is at the heart of much weirdness of LLMs. Do not brush it off.

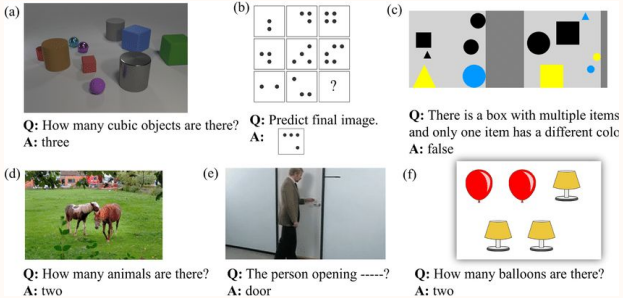
1. Why can't LLM spell words? Tokenization.
2. Why can't LLM do super simple string processing tasks like reversing a string? Tokenization.
3. Why is LLM worse at non-English languages (e.g. Japanese)? Tokenization.
4. Why is LLM bad at simple arithmetic? Tokenization.
5. Why did GPT-2 have more than necessary trouble coding in Python? Tokenization.
6. Why did my LLM abruptly halt when it sees the string "<|endoftext|>"? Tokenization.
7. What is this weird warning I get about a "trailing whitespace"? Tokenization.
8. Why the LLM break if I ask it about "SolidGoldMagikarp"? Tokenization.
9. Why should I prefer to use YAML over JSON with LLMs? Tokenization.
10. Why is LLM not actually end-to-end language modeling? Tokenization.
11. What is the real root of suffering? Tokenization.

LLM Training: Pretraining

Stage 1: Pretraining

- Involves compressing massive amounts of data from the internet
- Data is compressed into model parameters using thousands of **GPUs**, resulting in a large trained model file
- General Steps (happens once a year)
  - 1. Download ~10tb of text
  - 2. Get a cluster of ~6,000 gpus
  - 3. Compress text into neural network, pay ~$2M, wait ~12 days
  - 4. Obtain the base model
    - This base model cannot receive questions and answer them (yet)

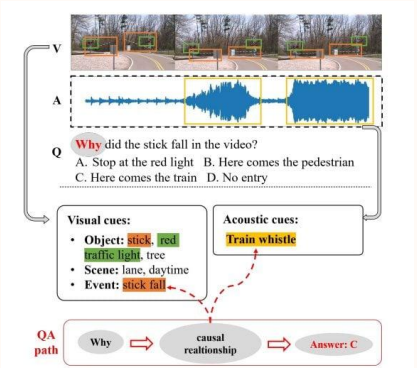
LLM Training: Fine Tuning



Stage 2: Fine Tuning

- To create the “assistant” model (model where you can ask questions and receive answers), you use fine tuning
  - Train the model with manually collected, high-quality Q&A datasets
  - Favor quality over quantity in terms of data
  - Trains AI to behave as a useful assistant, understanding how to answer questions outside the training set

LLM Training: Fine Tuning



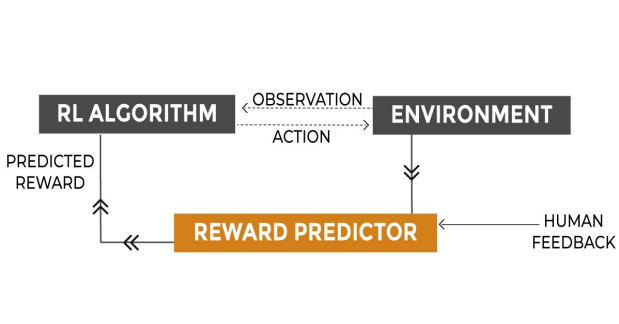
Stage 2: Fine Tuning

- General Steps (happens once a week)
  - 1. Write labeling instructions (guidelines for what constitutes as a “high-quality” response)
  - 2. Hire people, collect 100K high quality ideal Q&A responses, and/or comparisons
  - 3. Finetune base model on this data, wait ~1 day
  - 4. Obtain assistant model
  - 5. Run a lot of evaluations
  - 6. Deploy
  - 7. Monitor, collect misbehaviors, go back to step 1

LLM Training: Reinforcement Learning from Human Feedback

Stage 3 (Optional): Reinforcement Learning from Human Feedback (RLHF)

- This stage uses “comparison labels”, where human labelers rank multiple answers from an LLM from the same prompt to reinforce the more correct, accurate answer
- Sometimes easier and more efficient than a human producing Q&A datasets



Connecting to LLMs with Google Colab (Code)

https://colab.research.google.com/drive/1fdgFhicH92eZARYXiU__LA8ZHxS_8qd9?uspQ=sharing