

Large Language Model Applications

Spring 2026

Daphne Ippolito and Fernando Diaz

breaking down
“LLM application”



Define "LLM application"



Define “large language model”

What is a language model?

You might conclude that a likely word is blue, or green, or clear, but probably not refrigerator nor this. In this chapter we formalize this intuition by introducing n-gram **language models** or **LMs**. A language model is a machine learning model that predicts upcoming words. More formally, a language model assigns a **probability** to each possible next word, or equivalently gives a probability distribution over possible next words. Language models can also assign a probability to an entire sentence. Thus an LM could tell us that the following sequence has a much higher probability of appearing in a text:

all of a sudden I notice three guys standing on the sidewalk

than does this same set of words in a different order:

on guys all I of notice sidewalk three a sudden standing the

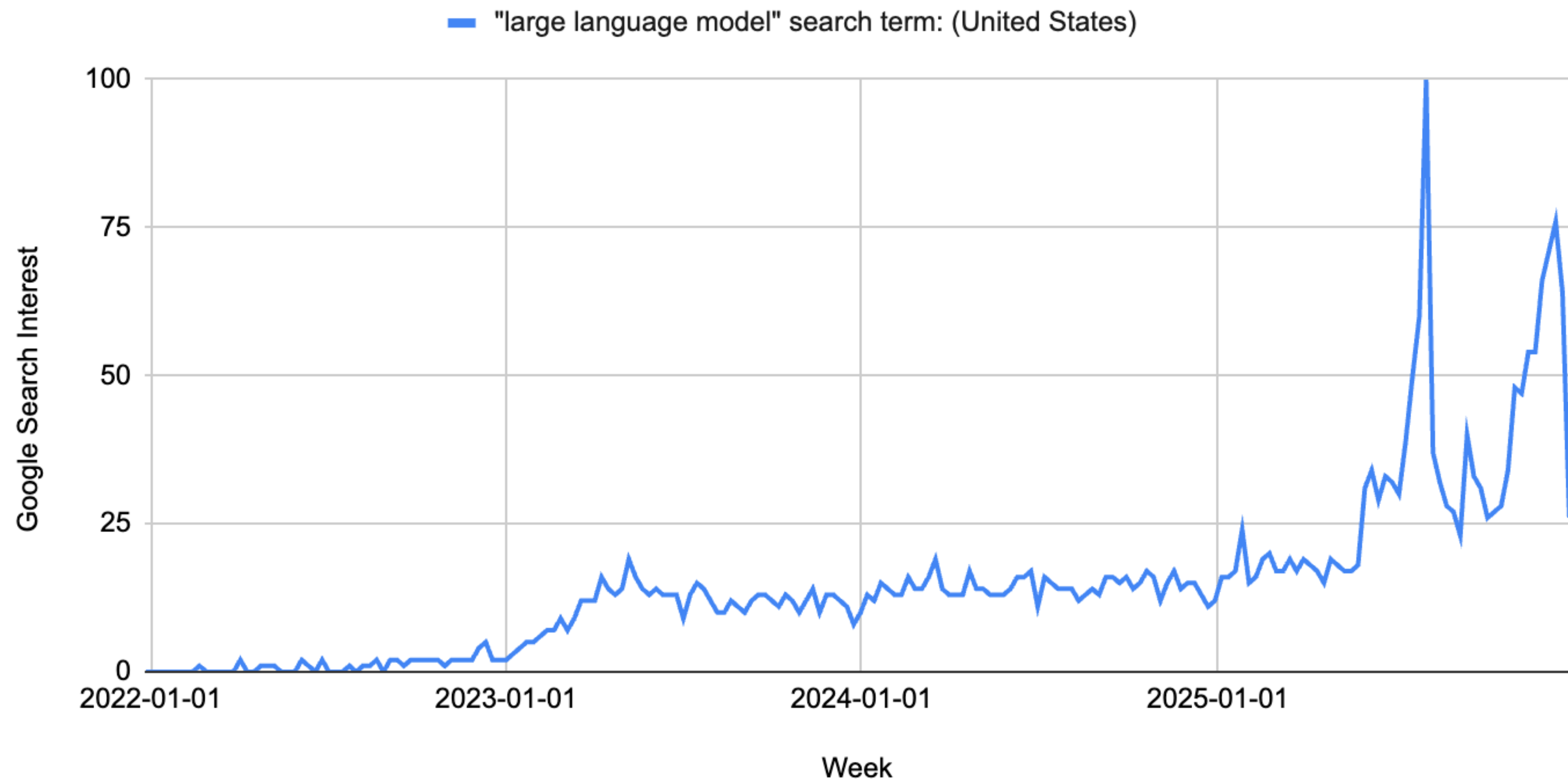
Why would we want to predict upcoming words? The main reason is that **large language models** are built just by training them to predict words!! As we'll see

How we'll deviate from this definition:

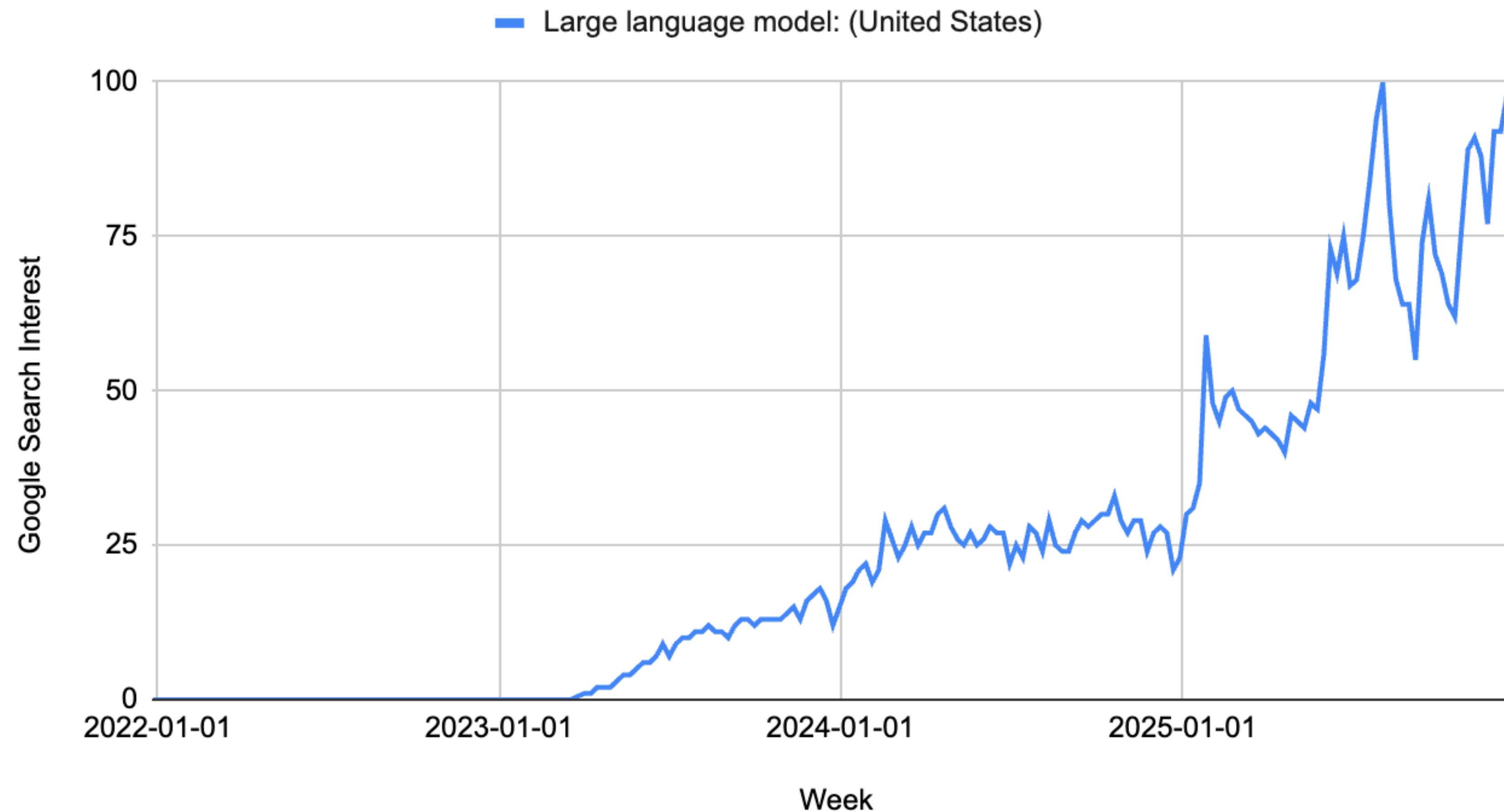
- Our class focuses on *neural* language models.
 - More specifically, we mostly focus on LMs that use the Transformer architecture.
- We'll take a very expansive definition of “language.” Words can be:
 - Code
 - Music
 - Proteins
 - Tool calls
 - Image patches
 - etc.

“large”

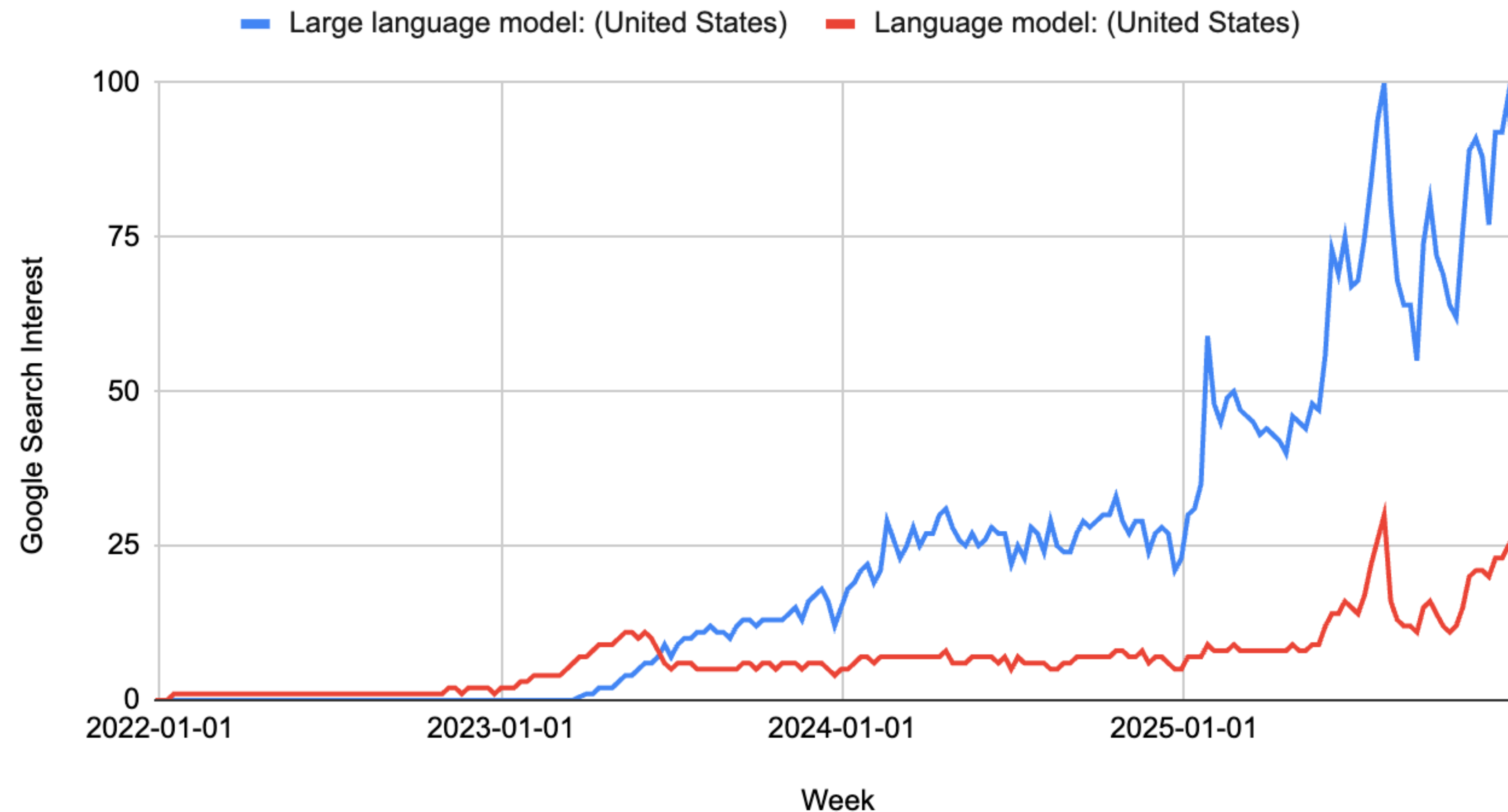
Google Search trends for LLM



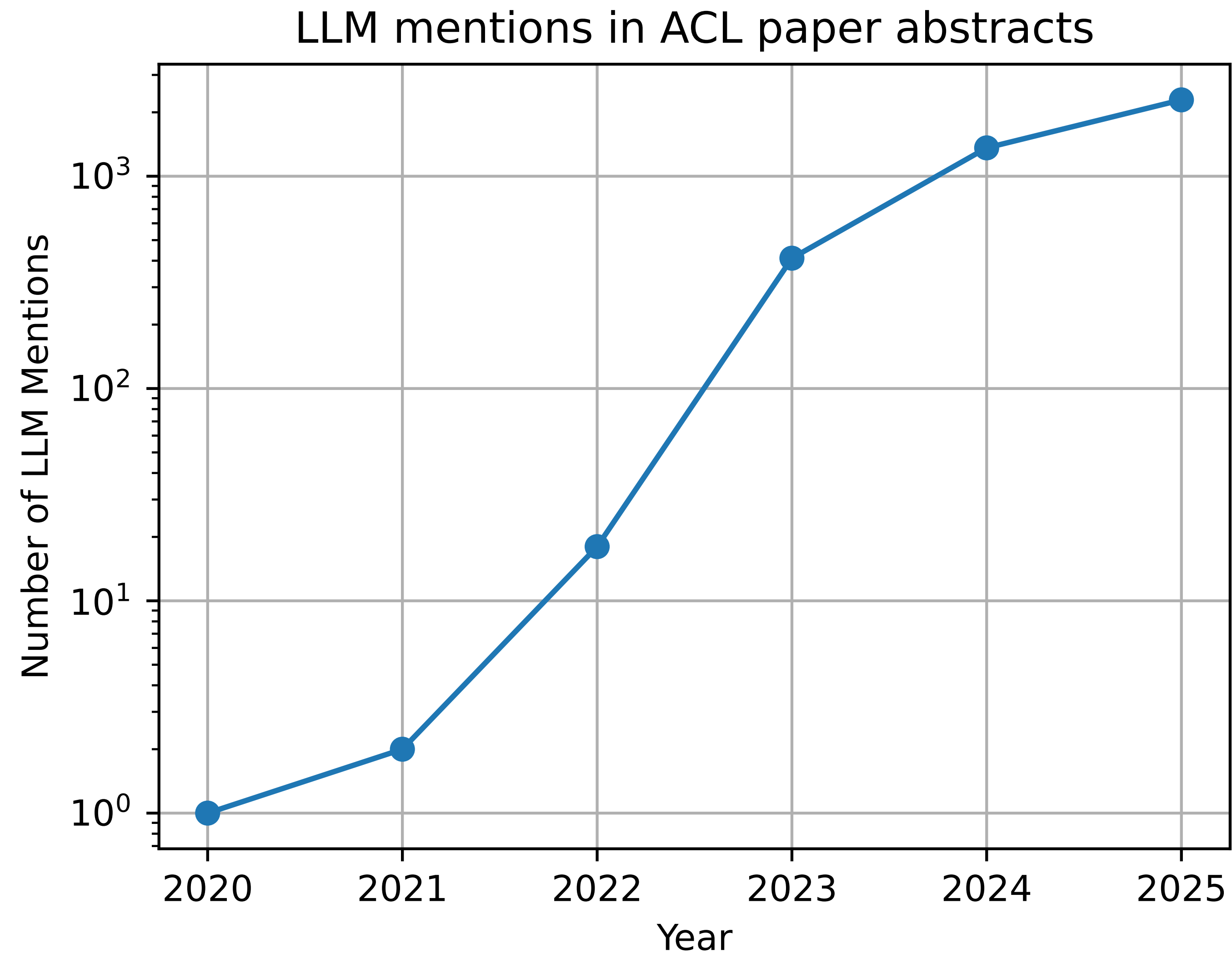
Google Search trends for LLM



Google Search trends for LLM vs. LM



LLMs in ACL Papers



The first LLM paper at ACL

 ACL Anthology [News](#) [FAQ](#) [Corrections](#) [Submissions](#) [Github](#)

Large Language Models in Machine Translation

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, Jeffrey Dean

Anthology ID: D07-1090

Volume: [Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning \(EMNLP-CoNLL\)](#)

Month: June

Year: 2007

Address: Prague, Czech Republic

Editor: [Jason Eisner](#)

Venues: [EMNLP](#) | [CoNLL](#)

SIGs: [SIGDAT](#) | [SIGNLL](#)

Publisher: Association for Computational Linguistics

Note: –

Pages: 858–867

Language: –

URL: <https://aclanthology.org/D07-1090/>

DOI: –

Bibkey: [brants-etal-2007-large](#)

Cite (ACL): Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. [Large Language Models in Machine Translation](#). In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867, Prague, Czech Republic. Association for Computational Linguistics. [📄](#)

Cite (Informal): [Large Language Models in Machine Translation](#) (Brants et al., EMNLP-CoNLL 2007) [📄](#)

Copy Citation: [📄 BibTeX](#) [📄 Markdown](#) [📄 MODS XML](#) [📄 Endnote](#) [More options...](#)

PDF: <https://aclanthology.org/D07-1090.pdf>

[📄 PDF](#)
[🗣️ Cite](#)
[🔍 Search](#)
[✎ Fix data](#)

The first LLM paper at ACL

“Large Language Models in Machine Translation”

Abstract

This paper reports on the benefits of large-scale statistical language modeling in machine translation. A distributed infrastructure is proposed which we use to train on up to 2 trillion tokens, resulting in language models having up to 300 billion n -grams. It is capable of providing smoothed probabilities for fast, single-pass decoding. We introduce a new smoothing method, dubbed *Stupid Backoff*, that is inexpensive to train on large data sets and approaches the quality of Kneser-Ney Smoothing as the amount of training data increases.

The first LLM paper at ACL

“Large Language Models in Machine Translation”

Abstract

This paper reports on the benefits of large-scale statistical language modeling in machine translation. A distributed infrastructure is proposed which we use to train on up to 2 trillion tokens, resulting in language models having up to 300 billion n -grams. It is capable of providing smoothed probabilities for fast, single-pass decoding. We introduce a new smoothing method, dubbed *Stupid Backoff*, that is inexpensive to train on large data sets and approaches the quality of Kneser-Ney Smoothing as the amount of training data increases.



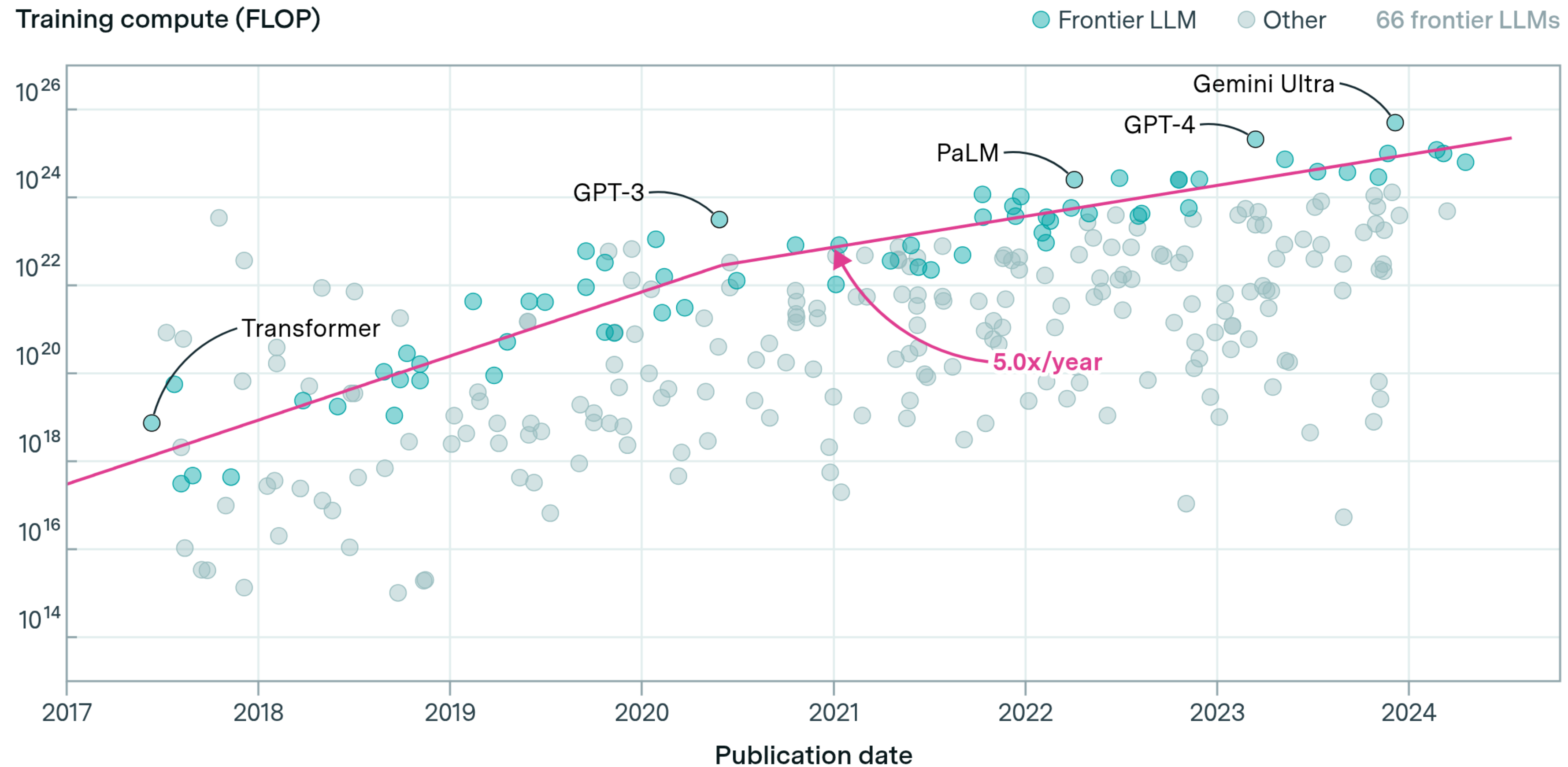
How is “large” measured today?

“Large” is a moving target

Training compute of frontier LLMs

EPOCH AI

Training compute (FLOP)



CC-BY

epoch.ai

“Training compute of frontier AI models grows by 4-5x per year.” [EpochAI report](#). 2024.

Language Models are Few-Shot Learners

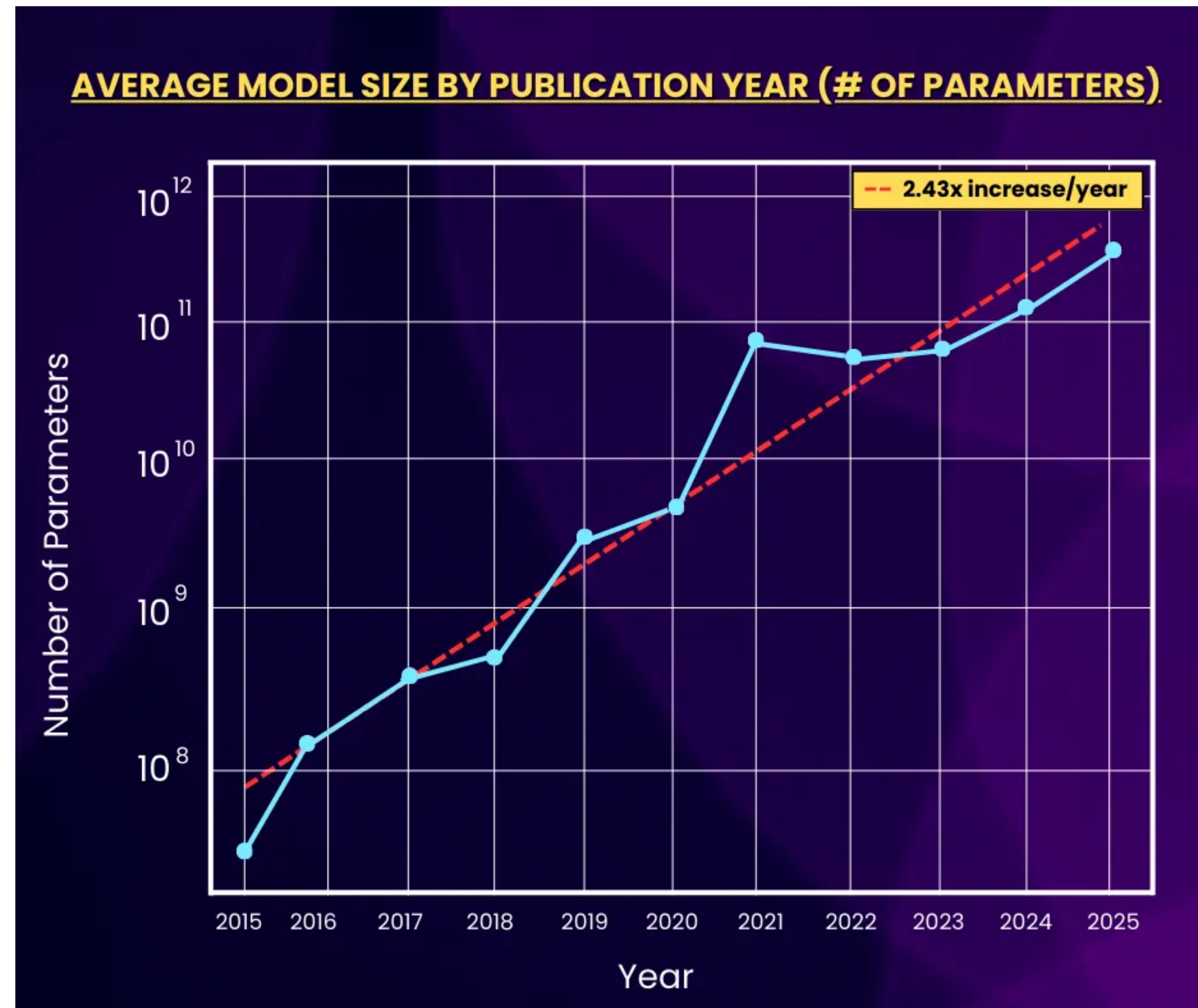
Tom B. Brown*	Benjamin Mann*	Nick Ryder*	Melanie Subbiah*	
Jared Kaplan [†]	Prafulla Dhariwal	Arvind Neelakantan	Pranav Shyam	Girish Sastry
Amanda Askell	Sandhini Agarwal	Ariel Herbert-Voss	Gretchen Krueger	Tom Henighan
Rewon Child	Aditya Ramesh	Daniel M. Ziegler	Jeffrey Wu	Clemens Winter
Christopher Hesse	Mark Chen	Eric Sigler	Mateusz Litwin	Scott Gray
Benjamin Chess	Jack Clark	Christopher Berner		
Sam McCandlish	Alec Radford	Ilya Sutskever	Dario Amodei	

OpenAI

Abstract

Recent work has demonstrated substantial gains on many NLP tasks and benchmarks by pre-training on a large corpus of text followed by fine-tuning on a specific task. While typically task-agnostic in architecture, this method still requires task-specific fine-tuning datasets of thousands or tens of thousands of examples. By contrast, humans can generally perform a new language task from only a few examples or from simple instructions – something which current NLP systems still largely struggle to do. Here we show that scaling up language models greatly improves task-agnostic, few-shot performance, sometimes even reaching competitiveness with prior state-of-the-art fine-tuning approaches. Specifically, we train GPT-3, an autoregressive language model with 175 billion parameters, 10x more than any previous non-sparse language model, and test its performance in the few-shot setting. For all tasks, GPT-3 is applied without any gradient updates or fine-tuning, with tasks and few-shot demonstrations specified purely via text interaction with the model. GPT-3 achieves strong performance on many NLP datasets, including translation, question-answering, and cloze tasks, as well as several tasks that require on-the-fly reasoning or domain adaptation, such as unscrambling words, using a novel word in a sentence, or performing 3-digit arithmetic. At the same time, we also identify some datasets where GPT-3’s few-shot learning still struggles, as well as some datasets where GPT-3 faces methodological issues related to training on large web corpora. Finally, we find that GPT-3 can generate samples of news articles which human evaluators have difficulty distinguishing from articles written by humans. We discuss broader societal impacts of this finding and of GPT-3 in general.

“Large” is a moving target



Measurements of largeness

- Amount of compute used in training
- Amount of data used in training
- Number of parameters in model
- Amount of compute used in inference?

How large is large enough to
count?

For this class...

“Large” simply mean large enough to solve meaningful problems.

This is a moving goalpost because as problems get solved, we find new challenging problems to apply language models to.

These new problems are often tackled by making bigger models.

LLM Applications

Components of an LLM Application

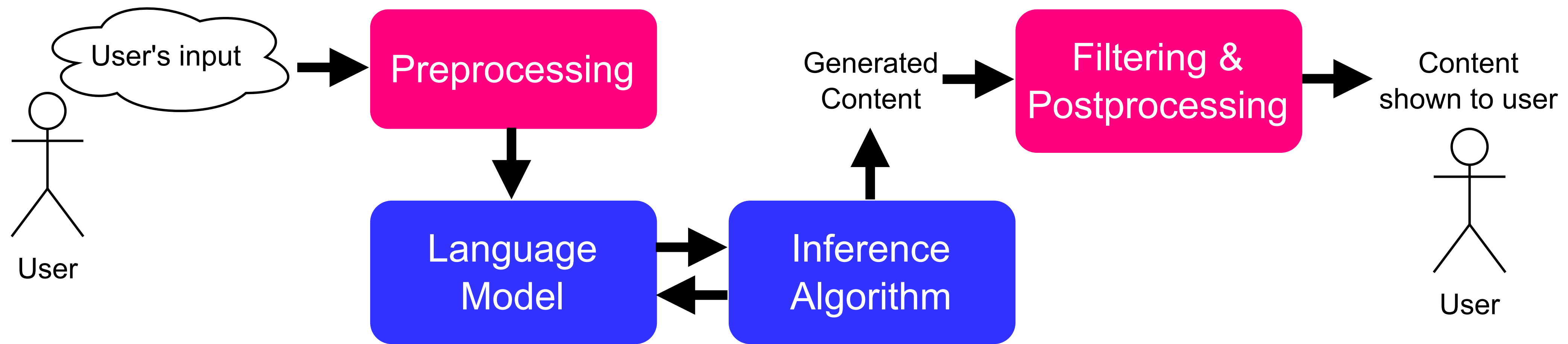
Language
Model

Components of an LLM Application

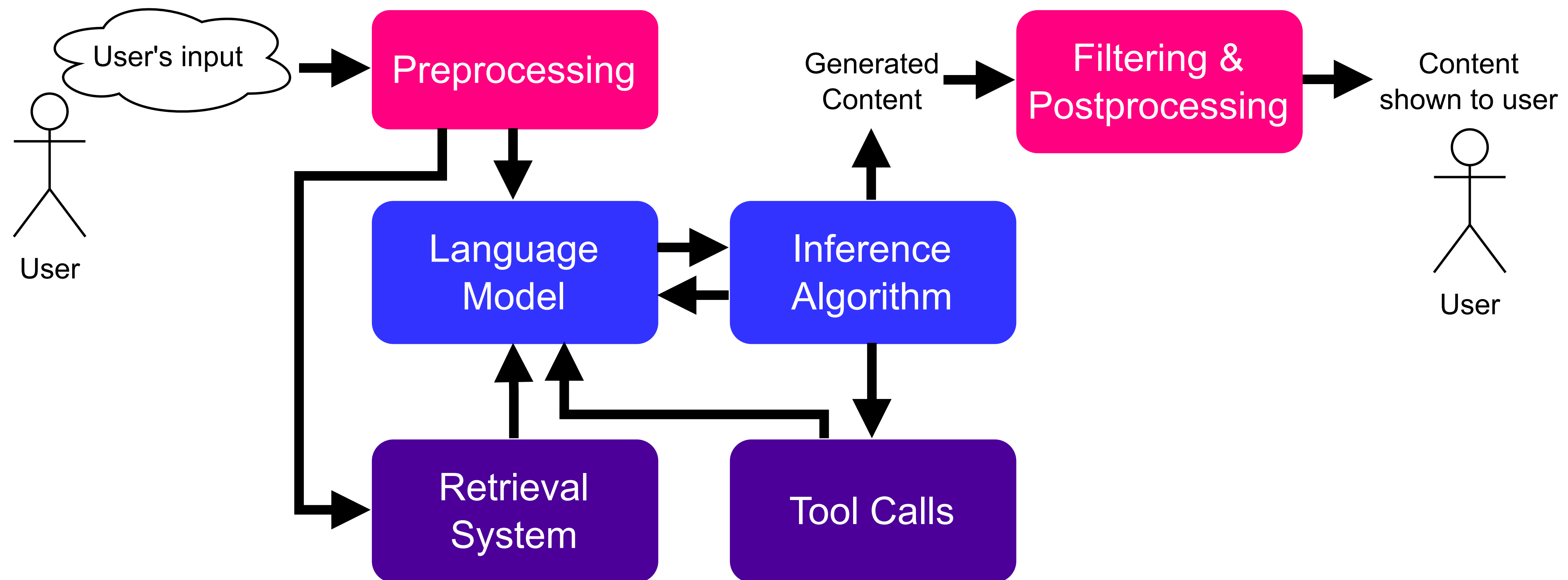
Language
Model

- Model architecture
- Training Data
 - Pre-training, mid-training, post-training, etc.
- Training objective
 - NLL, PPO, DPO, etc.

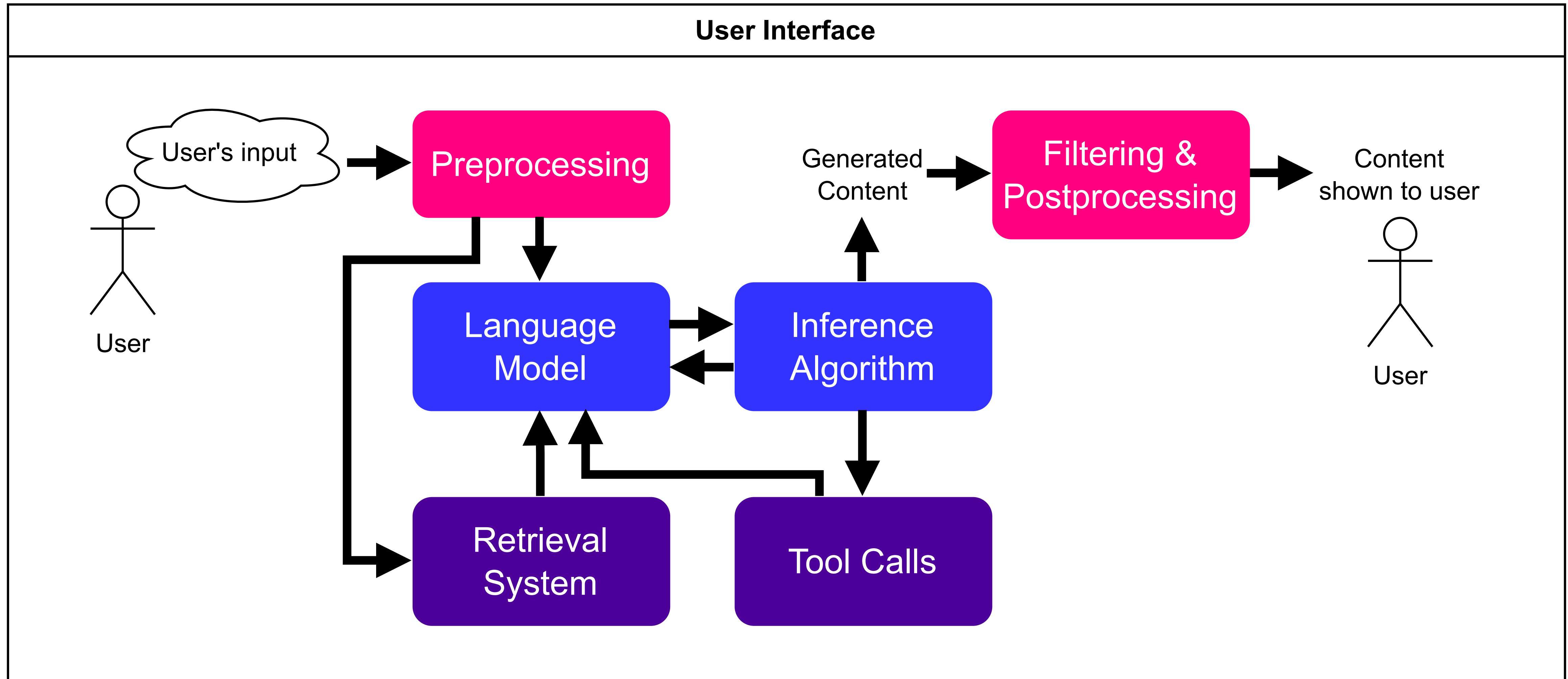
Components of an LLM Application



Components of an LLM Application



Components of an LLM Application



Natural Language Understanding vs. Natural Language Generation

(a pre LLM distinction)

Natural Language Understanding

circa 2000

- What does it mean for a computer system to understand natural language?
- Text-based applications:
 - finding appropriate documents from a database
 - information extraction
 - translation
 - summarization
- Dialog-based applications:
 - question-answering
 - automated customer service
 - spoken language control of machines

Natural Language Understanding

circa 2000

- Rule-based methods
 - Regular expressions, keyword search, etc.
- Machine learning classifier trained on handcrafted features
 - Text length, number of stopwords, n-gram frequencies, TF-IDF, etc.
 - Support vector machines, logistic regression, decision trees, etc.
- Linguistic Analysis
 - Parsing, linking syntax to semantics, formal knowledge representations

Natural Language Understanding

- Rule-based methods
 - Regular expressions, keyword search, etc.
- Machine learning classifier trained on handcrafted features
 - Text length, number of stopwords, n-gram frequencies, TF-IDF, etc.
 - Support vector machines, logistic regression, decision trees, etc.

Natural Language Generation

circa 2000

Natural Language Generation is the subfield of artificial intelligence and computational linguistics that focuses on computer systems that can produce understandable texts in English or other human languages.

~2000 definition of NLG

Natural Language Generation is the subfield of artificial intelligence and computational linguistics that focuses on computer systems that can produce understandable texts in English or other human languages.

NLU: language $\rightarrow$ data

NLG: data $\rightarrow$ language

NLG Research Questions (circa 2000)

1. How should computers interact with people?

- What is the best way for a machine to communicate information to a human?
- What kind of linguistic behaviour does a person expect of a computer they are communicating with, and how can this behaviour be implemented?

2. What constitutes 'readable' or 'appropriate' language in a given communicative situation?

- How can the appropriate pragmatic, semantic, syntactic, and psycholinguistic constraints be formalised?
- What role does context in its many aspects play in the choice of appropriate language?

3. How can typical computer representations of information - large amounts of low-level (often numeric) data - be converted into appropriate representations for humans?

- What types of domain and world models and associated reasoning are required to 'translate' information from computer representations to natural language?

NLG Research Questions (circa 2000)

1. How should computers interact with people?

- What is the best way for a machine to communicate information to a human?
- What kind of linguistic behaviour does a person expect of a computer they are communicating with, and how can this behaviour be implemented?

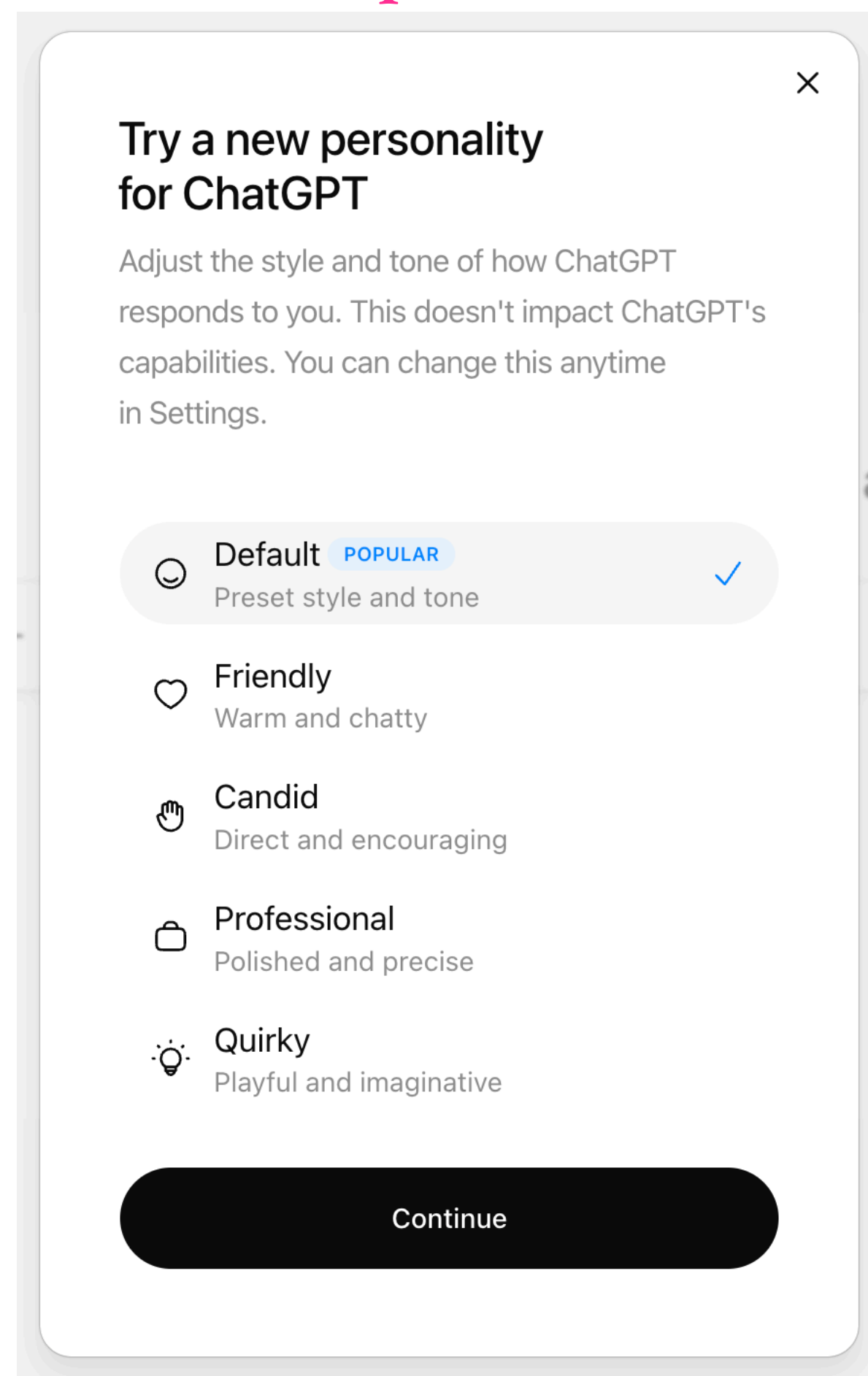
In 2025:

- Everything should be a chatbot implemented using an LLM
- Very little user-specific personalization

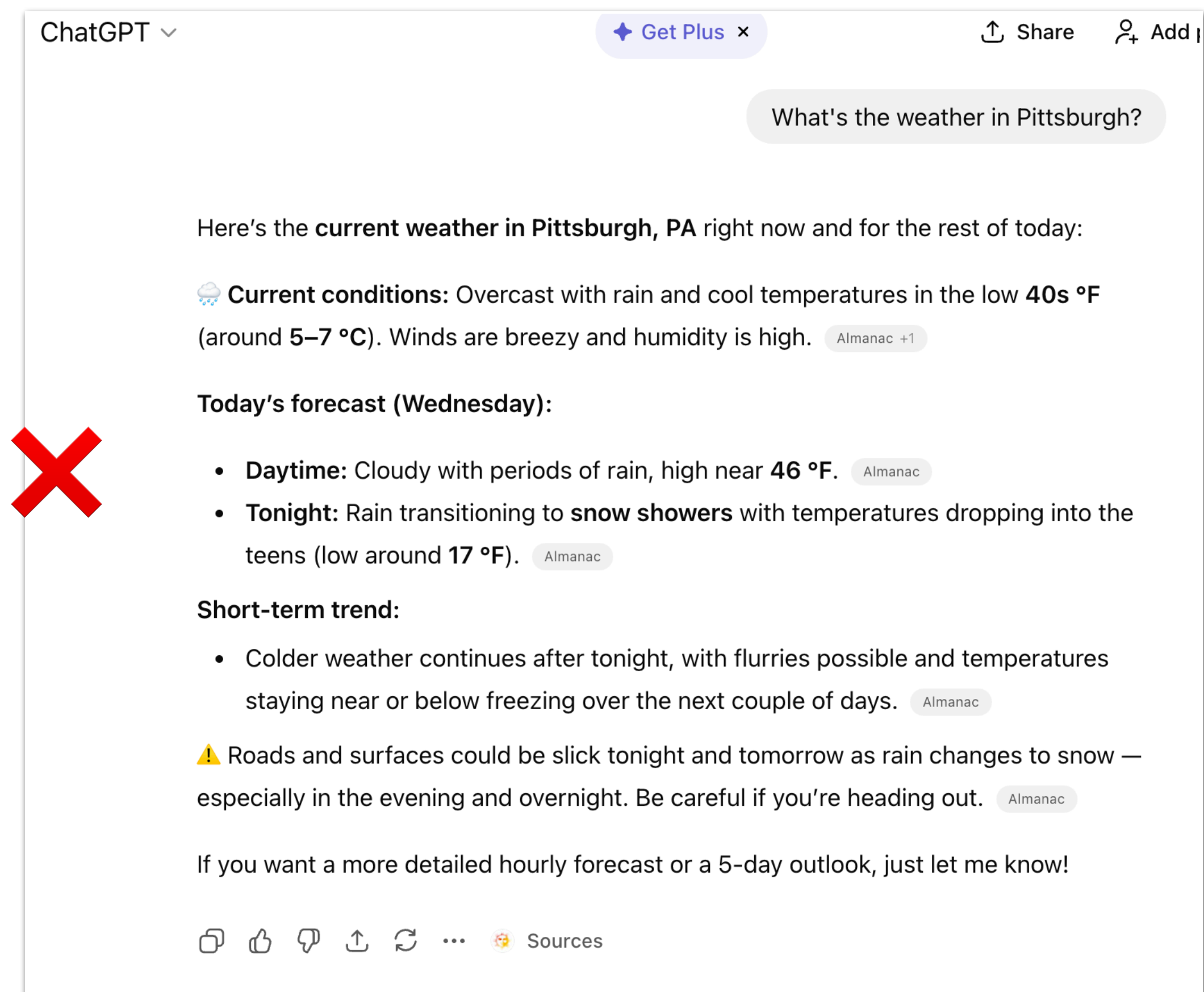
NLG Research Questions (circa 2000)

Things are changing

Growth in personalization



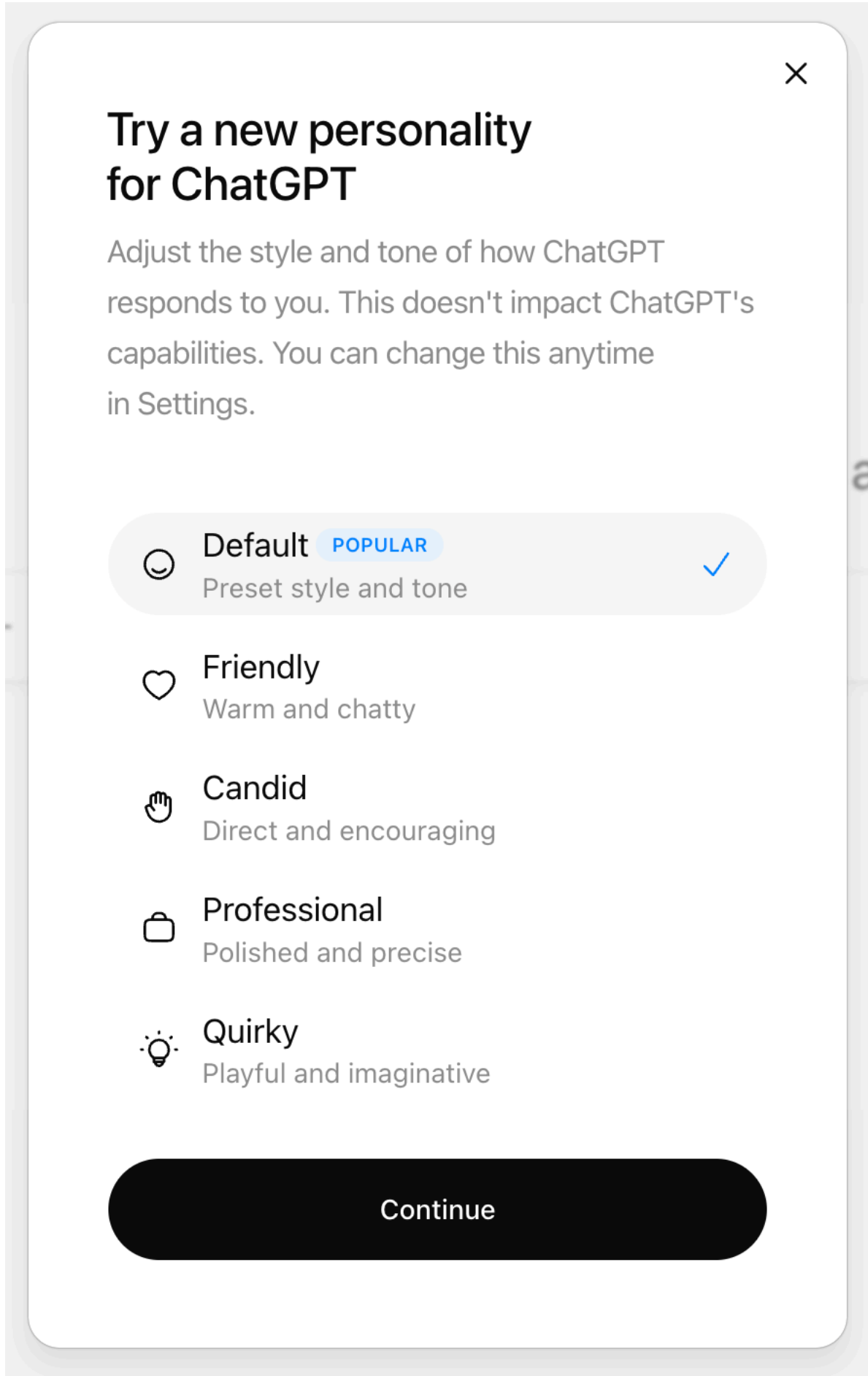
Movement away from chat



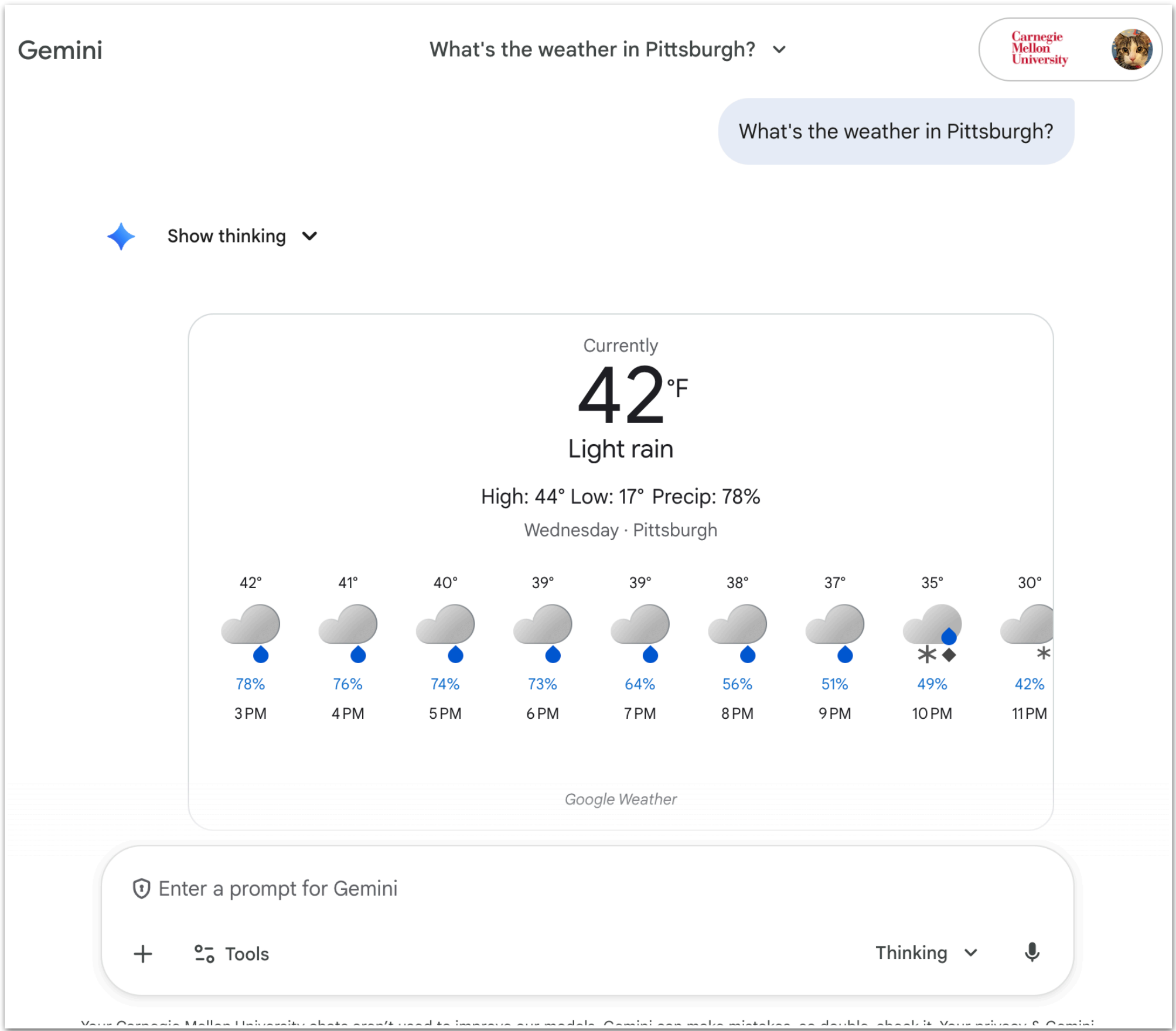
NLG Research Questions (circa 2000)

Things are changing

Growth in personalization



Movement away from chat



NLG Research Questions (circa 2000)

2. What constitutes 'readable' or 'appropriate' language in a given communicative situation?

- How can the appropriate pragmatic, semantic, syntactic, and psycholinguistic constraints be formalized?
- What role does context in its many aspects play in the choice of appropriate language?

In 2025:

- Chatbots should be “aligned,” but we don’t agree on what this means.
- Very little talk of formalizing the constraints put on LLM generations
- Incorporating context remains a challenge

NLG Research Questions (circa 2000)

3. How can typical computer representations of information - large amounts of low-level (often numeric) data - be converted into appropriate representations for humans?
- What types of domain and world models and associated reasoning are required to 'translate' information from computer representations to natural language?

In 2025:

- Organizing and communicating information remains a challenge
- High-quality code generation models are helping

How would you do it?



You're tasked with implementing a weather bot that sends a text message to users every morning with a summary of the weather they can expect for the day.

You have access to a REST API which returns a structured JSON of the weather forecast.

Should you use an LLM in your implementation?



You've been tasked by a small board game store to build an AI that goes through the purchasing catalog and suggests which games the store should stock.

You can look up information about a game on BoardGameGeek (review website for games).

You also have access to historical inventory data that lists games the store has ordered to its inventory and how long they took to sell.

Should you use an LLM in your implementation?

ways to use LMs besides just as
token generators

ways to use LMs besides just as token generators

probabilistic judges
feature extractors

Token Likelihoods

$$\Pr(w_i | w_s, w_I, \dots, w_{i-I})$$

Token Likelihoods

example task: detecting AI-generated text

A

The Monongahela River—often referred to locally as the Mon (/mɒn/)—is a 130-mile-long (210 km)[6] river on the Allegheny Plateau in north-central West Virginia and southwestern Pennsylvania. The river flows from the confluence of its west and east forks in north-central West Virginia northeasterly into southwestern Pennsylvania, then northerly to Pittsburgh and its confluence with the Allegheny River to form the Ohio River. The river's entire length is navigable via a series of locks and dams.

B

The Monongahela River is a tributary of the Ohio River, approximately 50 miles (80 km) long, in the U.S. state of West Virginia. It rises in the Allegheny Mountains of northeastern Pennsylvania and flows generally southward through the state, joining the Ohio River at Pittsburgh. The Monongahela River is the longest tributary of the Ohio River, and the only tributary to be navigable by large ships. The river is also the only navigable tributary of the Ohio River to be entirely within the state of West Virginia.

very unlikely ↔ very likely

Token Likelihoods

example task: classification

Context passed to model:

Answer the following question. Respond with just the letter of correct answer.

What's the name for fear of vegetables?

- A. Lachanophobia
- B. Gamophobia
- C. Lutraphobia
- D. Ephebiphobia

Compute

$$\arg \max_{a \in [A, B, C, D]} \Pr(a \mid \text{context})$$

Perplexity

Perplexity measures the normalized probability an LM assigns to a sequence of text.

$$\mathcal{P} = \exp \left(-\frac{1}{n} \sum_{i=1}^n \log f_{\theta}(x_i | x_1, \dots, x_{i-1}) \right)$$

- Used to assess the LM's confidence/surprisal at some piece of text.

Perplexity

example tasks:

- membership inference attack
- data filtering

Text Embeddings

much more on this in Jan 27 lecture

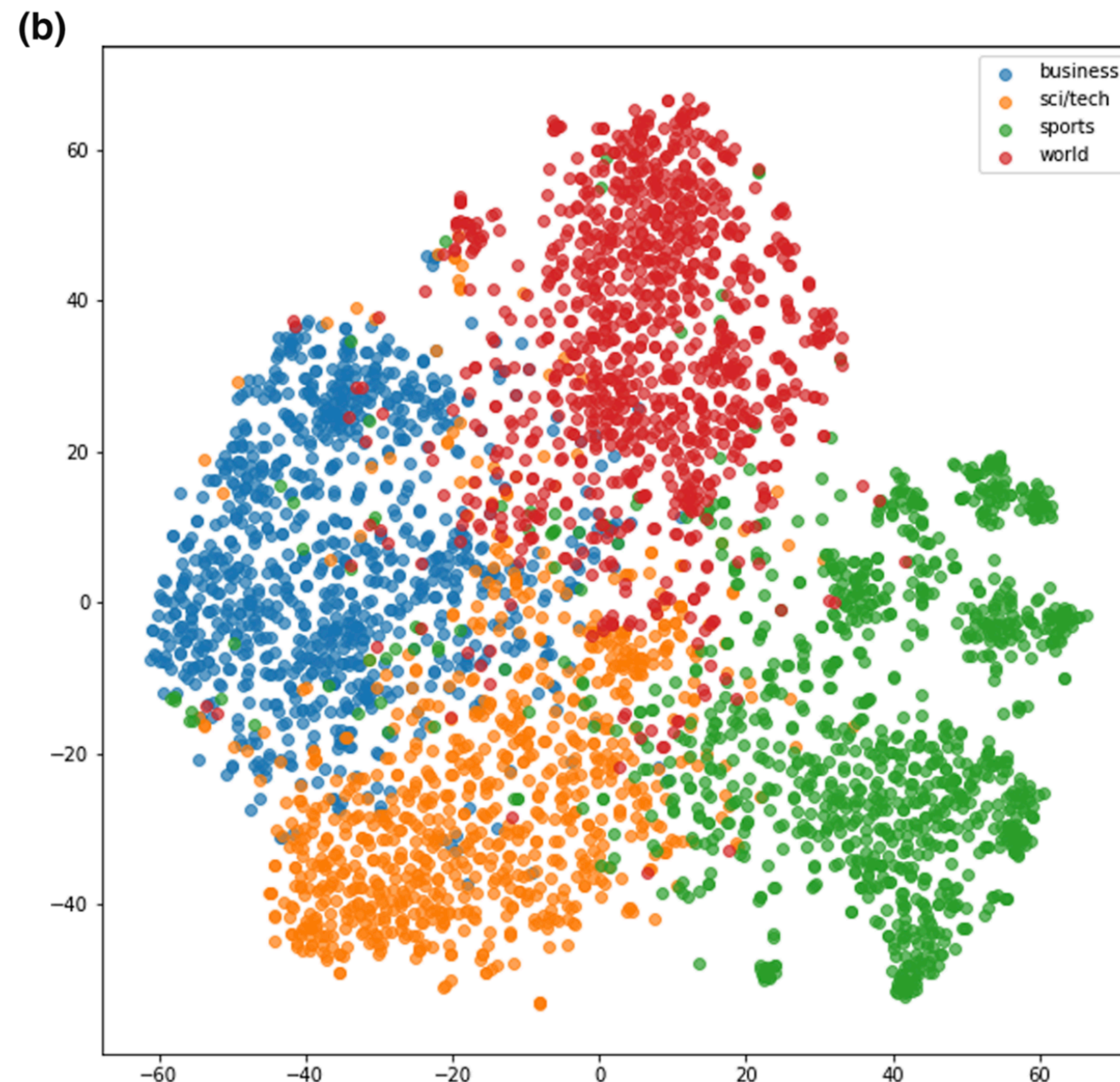


Fig. 3 t-SNE visualization of AG News ground truth label with text data representation from (a) TFIDF (b) BERT

Internal Activations

more in future lectures