

Programmatic use of large language models: A gentle introduction

XXXXXXXXXXXXXXXXXXXX

Dr. Jamie Cummins
University of Bern
Switzerland

About me

- Jamie (he/him) 🇮🇪
- PhD/postdoc background in psychology
- Working at Uni Bern since ~2023 as a senior postdoc in *meta-science*
- Tools and infrastructure for research quality management/trustworthiness assessment
- Large language models make up a big part of this
- My first time giving this workshop!

Today's workshop

- Aimed at *introducing* you all to interacting with large language models via the API
- Ideally we also work through some starting solutions for your research
- Some of you might find this very easy (already created API-based LLM workflows)
- Some of you might find this very hard (maybe less/no experience with code)
- The goal is generally to provide you with resources - if you need something, tell me!

Today's workshop agenda

- Introductory presentation + short break (1 hour total)
- Working through demo code + 2 breaks (2 hours total)
- Discussion of project use cases (~1 hour)

Introductory overview

- What are large language models and how do they work?
- What can LLMs do for researchers?
 - Automatically compare research materials
 - Screen titles and abstracts for a systematic review
 - Extract information from documents
- Why would we use LLMs *programmatically*?
- *How* do we use LLMs programmatically?
- Some considerations on the use of proprietary LLMs

What are large language models?

...it's kind of in the name!

deal with statistical prediction

What are large language models?

...it's kind of in the name!

the thing they are predicting is numeric representations of language

deal with statistical prediction

What are large language models?

...it's kind of in the name!

the thing they are predicting is numeric representations of language

they have *lots* of parameters!

deal with statistical prediction

What are large language models?

...it's kind of in the name!

How do we predict language?

- First, we need to figure out what *exactly* we're trying to predict
- The units of language we try to predict are called *tokens*
- the process of converting language to tokens is called *tokenization*
- So how do we tokenize?

How do we predict language?

- We could use **individual words** as tokens?
 - But this would be a huge corpus!
 - + poor generalization to infrequently used words

How do we predict language?

- We could use **individual words** as tokens?
 - But this would be a huge corpus!
 - + poor generalization to infrequently used words
- We could use **individual letters** to have fewer options?
 - But this would be very computationally intensive!
 - “Playing” -> 7 predictions
 - More room for error == more intensive training needed

So how do we predict language?

- Solution: we use *word chunks*
 - Break existing words down, but not letter by letter
 - e.g.: “playing” becomes “play” and “ing”
 - Chunks are created based on most common occurrences in the corpus
 - The method used by OpenAI etc. is called “byte-pair encoding”
 - Also encodes some additional information (e.g., spaces)

So how do we predict language?

“There once was a wondrous cat who sat on a beautiful ”



```
There once was a wondrous cat who sat on a beautiful
```



```
[5632, 4730, 673, 261, 159740, 45170, 9059, 1218, 10139, 402, 261, 6107,  
220]
```

So how do we predict language?

“There there There there there.”



There there There there there.



[5632, 1354, 3274, 1354, 1354, 13]

So how do we predict language?

“There once was a wondrous cat who sat on a beautiful ”

```
[5632, 4730, 673, 261, 159740, 45170, 9059, 1218, 10139, 402, 261, 6107,  
220]
```

These are our inputs.

How do we predict the output (aka, the next token)?

We also need numeric representations of the
semantic content each of these tokens!

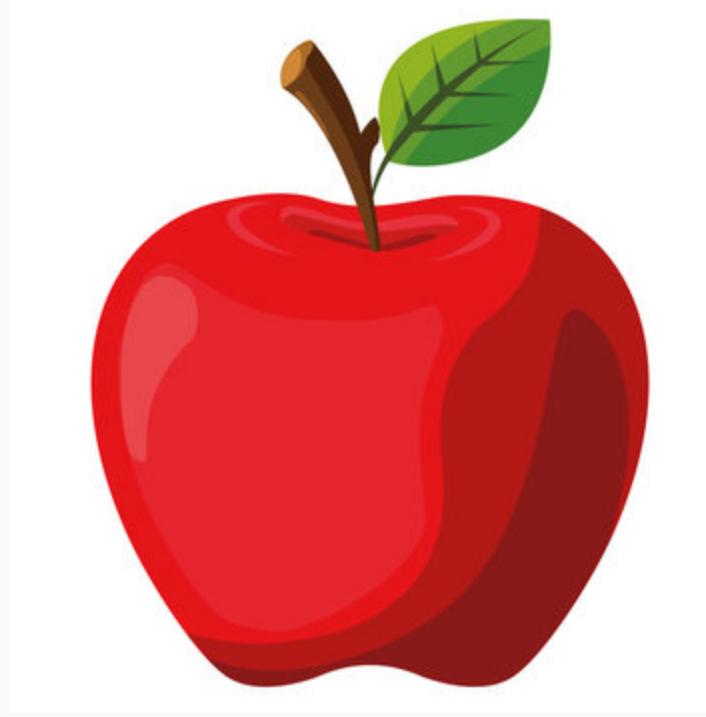
We are dealing with **numeric representations**
of semantic information
dependent on **context**.

These representations are called *embeddings*.

```
[5632, 4730, 673, 261, 159740, 45170, 9059, 1218, 10139, 402, 261, 6107,  
220]
```

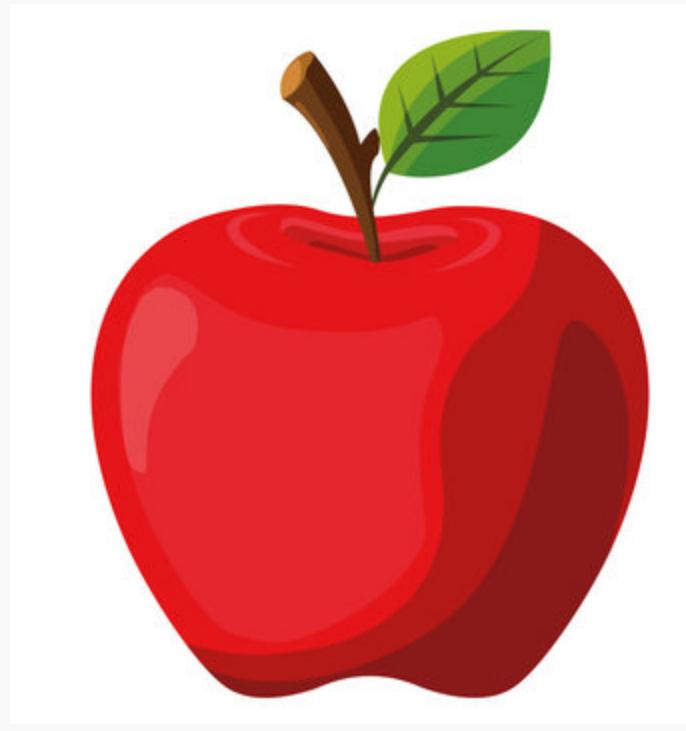
Token ID	dim₁	dim₂	dim₃	dim₄	dim₅	dim₆	dim₇	dim₈
5632	0.27	0.17	0.39	0.05	0.17	0.58	0.27	0.22
4730	0.45	0.15	0.29	0.96	0.59	0.79	0.56	0.22
673	0.68	0.30	0.30	0.96	0.96	0.59	0.37	0.19
261	0.01	0.57	0.06	0.46	0.33	0.34	0.87	0.96
159740	0.75	0.22	0.29	0.50	0.13	0.25	0.81	0.53
45170	0.51	0.94	0.49	0.87	0.78	0.92	0.16	0.07
9059	0.04	0.80	0.58	0.97	0.28	0.34	0.95	0.46
1218	0.85	0.40	0.73	0.95	0.86	0.77	0.06	0.32
10139	0.68	0.15	0.02	0.62	0.56	0.99	0.82	0.94
402	0.50	0.62	0.64	0.34	0.43	0.11	0.76	0.90
6107	0.29	0.61	0.84	0.47	0.41	0.65	0.50	0.93
220	0.36	0.70	0.51	0.88	0.10	0.46	0.33	0.05

Wait, what the hell are these numbers?



How might we describe these objects numerically?





Object	Red	Green	Blue	Width	Length
Apple	1	0	0	1	0.8
Banana	0.5	0.5	0	0.8	1

The key difference is that embeddings vectors are *induced* through machine learning.

We cannot simply interpret embeddings vector dimensions or immediately understand what is represented in a given dimension.

But they are *useful* for prediction!

[5632, 4730, 673, 261, 159740, 45170, 9059, 1218, 10139, 402, 261, 6107,
220]

Token ID	dim ₁	dim ₂	dim ₃	dim ₄	dim ₅	dim ₆	dim ₇	dim ₈
5632	0.27	0.17	0.39	0.05	0.17	0.58	0.27	0.22
4730	0.45	0.15	0.29	0.96	0.59	0.79	0.56	0.22
673	0.68	0.30	0.30	0.96	0.96	0.59	0.37	0.19
261	0.01	0.57	0.06	0.46	0.33	0.34	0.87	0.96
159740	0.75	0.22	0.29	0.50	0.13	0.25	0.81	0.53
45170	0.51	0.94	0.49	0.87	0.78	0.92	0.16	0.07
9059	0.04	0.80	0.58	0.97	0.28	0.34	0.95	0.46
1218	0.85	0.40	0.73	0.95	0.86	0.77	0.06	0.32
10139	0.68	0.15	0.02	0.62	0.56	0.99	0.82	0.94
402	0.50	0.62	0.64	0.34	0.43	0.11	0.76	0.90
6107	0.29	0.61	0.84	0.47	0.41	0.65	0.50	0.93
220	0.36	0.70	0.51	0.88	0.10	0.46	0.33	0.05

What's missing here?

[5632, 4730, 673, 261, 159740, 45170, 9059, 1218, 10139, 402, 261, 6107,
220]

Token ID	dim ₁	dim ₂	dim ₃	dim ₄	dim ₅	dim ₆	dim ₇	dim ₈
5632	0.27	0.17	0.39	0.05	0.17	0.58	0.27	0.22
4730	0.45	0.15	0.29	0.96	0.59	0.79	0.56	0.22
673	0.68	0.30	0.30	0.96	0.96	0.59	0.37	0.19
261	0.01	0.57	0.06	0.46	0.33	0.34	0.87	0.96
159740	0.75	0.22	0.29	0.50	0.13	0.25	0.81	0.53
45170	0.51	0.94	0.49	0.87	0.78	0.92	0.16	0.07
9059	0.04	0.80	0.58	0.97	0.28	0.34	0.95	0.46
1218	0.85	0.40	0.73	0.95	0.86	0.77	0.06	0.32
10139	0.68	0.15	0.02	0.62	0.56	0.99	0.82	0.94
402	0.50	0.62	0.64	0.34	0.43	0.11	0.76	0.90
6107	0.29	0.61	0.84	0.47	0.41	0.65	0.50	0.93
220	0.36	0.70	0.51	0.88	0.10	0.46	0.33	0.05

Position in the sentence!

So we also assign *positional embeddings* separately...

Token ID	dim ₁	dim ₂	dim ₃	dim ₄	dim ₅	dim ₆	dim ₇	dim ₈
5632	0.27	0.17	0.39	0.05	0.17	0.58	0.27	0.22
4730	0.45	0.15	0.29	0.96	0.59	0.79	0.56	0.22
673	0.68	0.30	0.30	0.96	0.96	0.59	0.37	0.19
261	0.01	0.57	0.06	0.46	0.33	0.34	0.87	0.96
159740	0.75	0.22	0.29	0.50	0.13	0.25	0.81	0.53
45170	0.51	0.94	0.49	0.87	0.78	0.92	0.16	0.07
9059	0.04	0.80	0.58	0.97	0.28	0.34	0.95	0.46
1218	0.85	0.40	0.73	0.95	0.86	0.77	0.06	0.32
10139	0.68	0.15	0.02	0.62	0.56	0.99	0.82	0.94
402	0.50	0.62	0.64	0.34	0.43	0.11	0.76	0.90
6107	0.29	0.61	0.84	0.47	0.41	0.65	0.50	0.93
220	0.36	0.70	0.51	0.88	0.10	0.46	0.33	0.05

	dim ₁	dim ₂	dim ₃	dim ₄	dim ₅	dim ₆	dim ₇	dim ₈
Position 0	0.96	0.7	1.0	0.22	0.36	0.74	1.0	0.32
Position 1	0.14	0.38	0.32	0.37	0.71	0.9	0.53	0.25
Position 2	0.67	0.56	0.54	0.89	0.84	0.31	0.63	0.68
Position 3	0.97	0.89	0.94	0.64	0.61	0.23	0.49	0.81
Position 4	0.84	0.53	0.76	0.5	0.85	0.62	0.86	0.23

The input that the model receives is the sum of the embedding and positional vectors.

Then a bunch of other stuff happens...

where the magic really happens

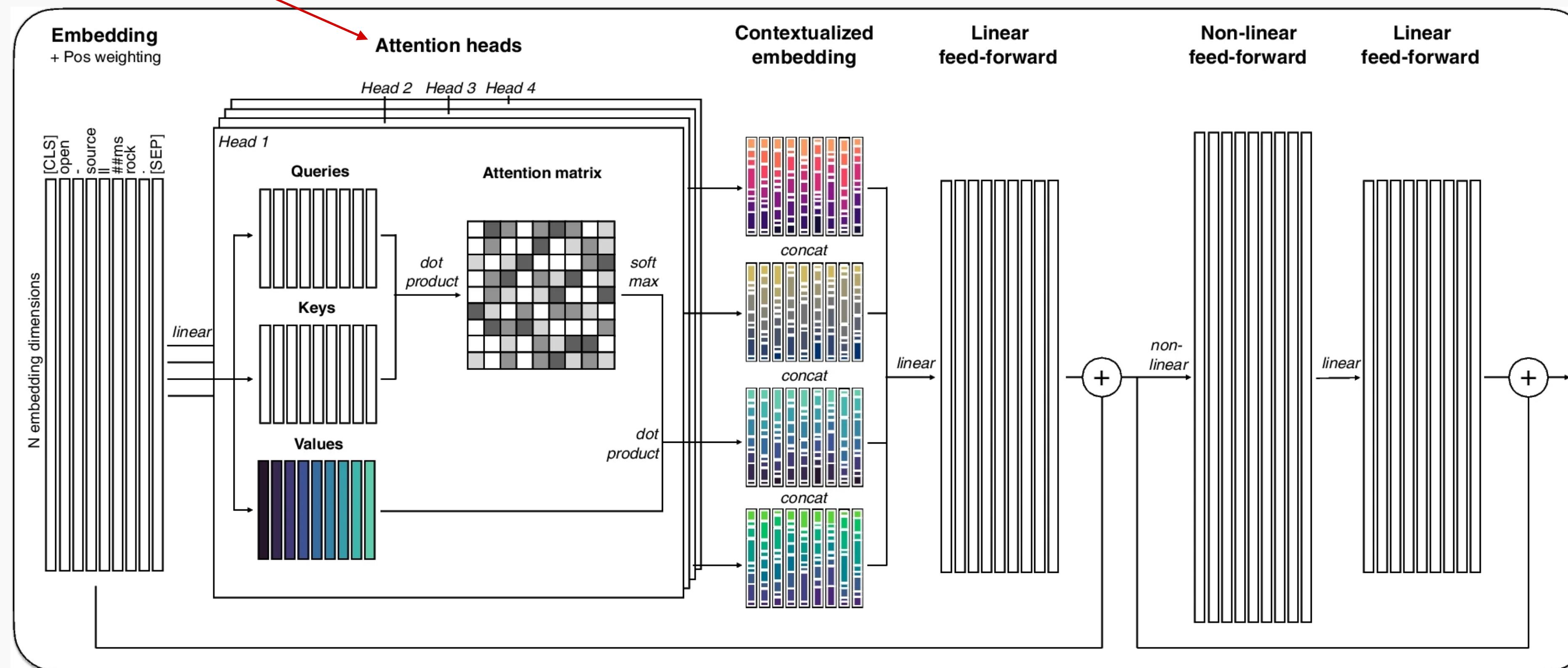


Figure from Hussain et al. (2024). A tutorial on open-source large language models for behavioral science.

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

After all of this, we end up with logit predictions
for *every single possible token*.

Token	Logit	Probability
mat	6.0	0.248
creature	5.5	0.151
day	5.0	0.091
flower	4.5	0.055
dragon	4.0	0.034
.	3.5	0.02

Then, a sampling strategy is applied over this distribution for the prediction.

When we use ChatGPT or other LLMs for inference,
this process is just repeated for *every token*.

Input 1:

“There once was a wondrous cat who sat on a beautiful ”

Output 1:

“mat”

Input 2:

“There once was a wondrous cat who sat on a beautiful mat”

Output 2:

“. ”

Input 3:

“There once was a wondrous cat who sat on a beautiful mat.”

Output 3:

“One ”

Input 4:

“There once was a wondrous cat who sat on a beautiful mat. One ”

Output 4:

“day ”

So the input for the LLM is the entirety* of the previous conversation!

*not always but we'll get to that

Hi there! My name is Jamie. What's your name?

Hi Jamie! I'm ChatGPT—but you can just call me Chat if you like. Nice to meet you. How can I help today?

I'd prefer not to do that. Sorry. We're just not that close.

Totally fair, Jamie. Formal it is—ChatGPT it shall remain. What can I do for you today?

Can you give yourself an even more formal name?

Certainly, Dr. Cummins.

You may refer to me as **C.G.P.T., Esq.**, a computational gentleman of probabilistic tendencies, at your service.

Shall we proceed with business?

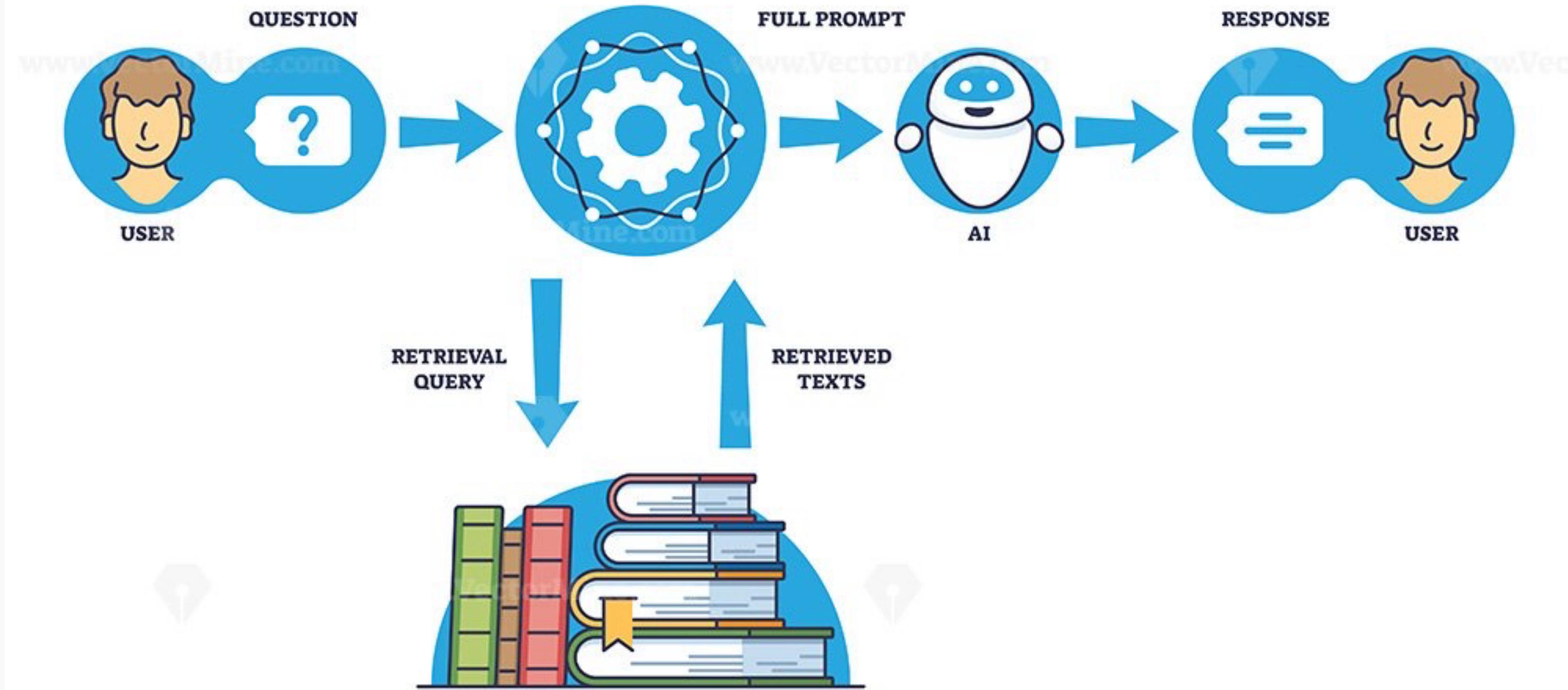
For this reason, letting a model give more output before responding leads to better output on average.

This is how the *reasoning* family of models work!

**But what about with very long documents
or very long conversations?**

RAG

RETRIEVAL AUGMENTED GENERATION



You can embed text of any length - not just single tokens.

Like paragraphs, or text chunks, or *prompts*...

“The sample size, which is related to the number of participants which are reported in the manuscript.”

```
          0  
0  0.023441  
1 -0.013284  
2  0.041531  
3 -0.003225  
4 -0.004297  
... ...  
1531 -0.005362  
1532 -0.016538  
1533 -0.001271  
1534  0.009422  
1535  0.019897  
  
[1536 rows x 1 columns]
```

OSC (2015). Estimating the Reproducibility of Psychological Science

	chunk	embedding
0	Title: Estimating the reproducibility of psych...	[0.050331443548202515, 0.01178510021418333, 0....
1	Thirty-six percent of replications had signifi...	[0.04425464943051338, -0.0023916689679026604, ...
2	This project provides accumulating evidence fo...	[0.04153410717844963, -0.0018653592560440302, ...
3	Direct replication is the attempt to recreate ...	[0.036356888711452484, 0.0043677156791090965, ...
4	Potentially problematic practices include sele...	[0.040376920253038406, 0.010345174930989742, 0....
5	The units of analysis for inferences about rep...	[0.016850758343935013, 0.0059136622585356236, ...
6	Project coordinators facilitated matching arti...	[0.028508516028523445, 0.0037176646292209625, ...
7	The key result had to be represented as a sing...	[0.02611692063510418, 0.019338591024279594, 0....
8	The most common reasons for failure to match a...	[0.016962390393018723, 0.028942249715328217, 0....
9	These included characteristics of the original...	[0.027320412918925285, 0.005955172702670097, 0....
10	From this pool of available studies, replicati...	[0.03738851100206375, -0.004830995574593544, 0....
11	Using only the nonsignificant P values of the ...	[0.021562064066529274, -0.03394116833806038, 0....
12	Third, we computed the proportion of study-pai...	[0.04071827977895737, -0.01888895407319069, 0....
13	Exclusions (explanation provided in supplement...)	[0.02814759872853756, -0.015499881468713284, 0....
14	Standard errors could only be computed if test...	[0.03007425181567669, -0.007572266738861799, 0....
15	Analysis of moderators We correlated the five ...	[0.02257361076772213, 0.008557644672691822, 0....
16	When there is no effect to detect, the null di...	[0.026772739365696907, -0.023849690333008766, ...
17	Also, the replication "succeeds" when the resu...	[0.057230137288570404, -0.023440003395080566, ...

We can then use similarity metrics on these N-dimensional vectors

Most commonly cosine similarity, which is computed as:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

Then we can query the top k matches of our embedded document against the prompt embeddings and end up with:

		chunk similarity
6	Project coordinators facilitated matching arti...	0.470970
12	Third, we computed the proportion of study-pai...	0.447195
13	Exclusions (explanation provided in supplement...	0.440072
9	These included characteristics of the original...	0.429687
8	The most common reasons for failure to match a...	0.425096

Now we feed *only these paragraphs* into the prompt the model sees!

What can LLMs do for researchers?

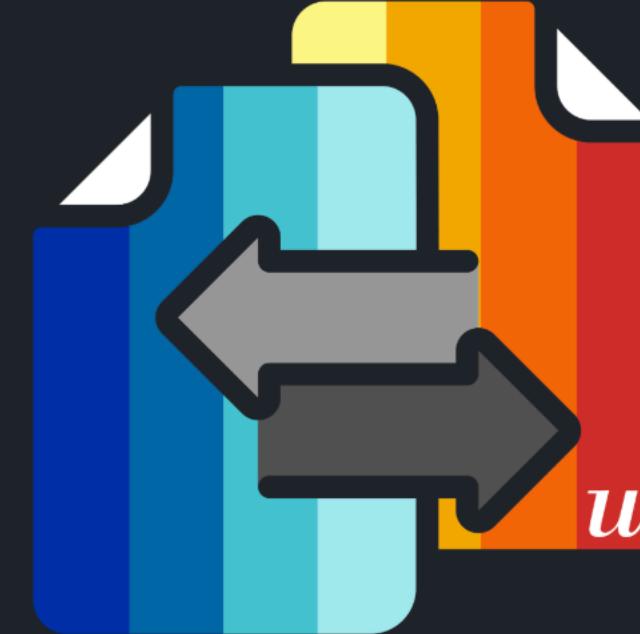
...a lot!

(maybe)

Automatically compare research materials

RegCheck 

Home Demo Tools ▾ Team Jobs Contact



Compare preregistrations with papers.
Instantly.

RegCheck leverages large-language models and AI to automatically compare preregistration plans with scientific publications. This lets researchers effortlessly identify if and how executed studies deviated from the initial plan.

Try it now

Your output. Simplified.

A check of Experiment 2 from [Hughes, Cummins & Hussey, 2023](#).

[View Prereg](#) [View Paper](#) [Download as CSV](#)

Hypothesis	Study Feature	Information in paper	Preregistered Protocol	Match**
M1: AMP effect will be demonstrated	Sample size before exclusions	214	150	TRUE
M1: AMP effect will be demonstrated	Sample size after exclusions	147	150	FALSE
M1: AMP effect will be demonstrated	Planned variable(s) to measure/test	Valence Ratings (pleasant or unpleasant) of the target stimulus	Evaluations within the AMP as pleasant or unpleasant	TRUE
M1: AMP effect will be demonstrated	Planned processing/scoring	Model acknowledges non-independence of multiple data points; modeled trial type as	Frequentist logistic mixed-effects model; include subject ID as random intercept	TRUE

<https://regcheck.app>

Screen titles and abstracts for a systematic review

	id	title	abstract	final_decision	title_and_abstract_screening_ai_linebyline_unblinded	title_and_abstract_screening_ai_linebyline_blinded
	0 1	Factors Influencing Players' Susceptibility to...	Social gaming networks have become a new pheno...	0	1	1
	1 2	The Influence of Privacy and Trust on the Adop...	In Bangladesh, the expansion of Internet banki...	1	1	1
	2 3	An Examination of Gen Z Learners Attending a M...	This paper presents the preliminary findings o...	0	1	1
	3 4	Factors Affecting Security Behavior of Kenyan ...	The purpose of this study was to identify the ...	1	1	1
	4 5	Teamwork attitude, interest, and self-efficacy...	The complex and multidisciplinary nature of to...	0	0	0
	5 6	Instruction in 802.11 Technology in Online Vir...	A novel approach to remote instruction in 802....	0	0	1
	6 7	Information security professional perceptions ...	Knowledge sharing is an important component of...	0	1	1
	7 8	Information security awareness behavior among ...	The exchange of information is a key factor in...	1	1	1

Extract information from documents

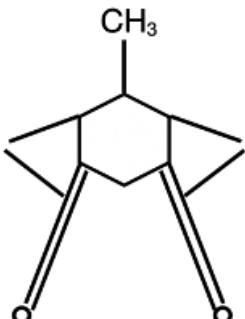
Posturogenic Dominatrone

Therapeutic Function: Stimulant

Chemical Name: Alpha-Tetraposturone Hydrodominate Sulfate

Common Name: Power posing

Structural Formula:



Chemical Abstracts Registry No.: 284739-39384

Trade Name	Manufacturer	Country	Year Introduced
Gravitasol	Stature Sciences	USA	2010
Akimbolin	Dominion Biotech	Switz.	2018

Raw Materials

Bi-angular Upper Limb Configuration

Epistemic Affirmation Receptivity

Manufacturing Process

The manufacturing process for **Posturogenic Dominatrone** begins with the synthesis of benzene-derived confidence stabilizers, which are carefully heated to sublimation under conditions of pure ambition. These compounds are then subjected to high-pressure akimbo infusion, where linear bonds are meticulously aligned to emulate the structural geometry of a dominant pose. The resulting molecular blend is cooled in a wide-stance polymer chamber, where it crystallizes into a form optimized for self-assured bioavailability. The final compound undergoes spectroscopic gravitas validation before being encapsulated in charisma-sealed blister packs, ensuring maximum potency for every dose.

References

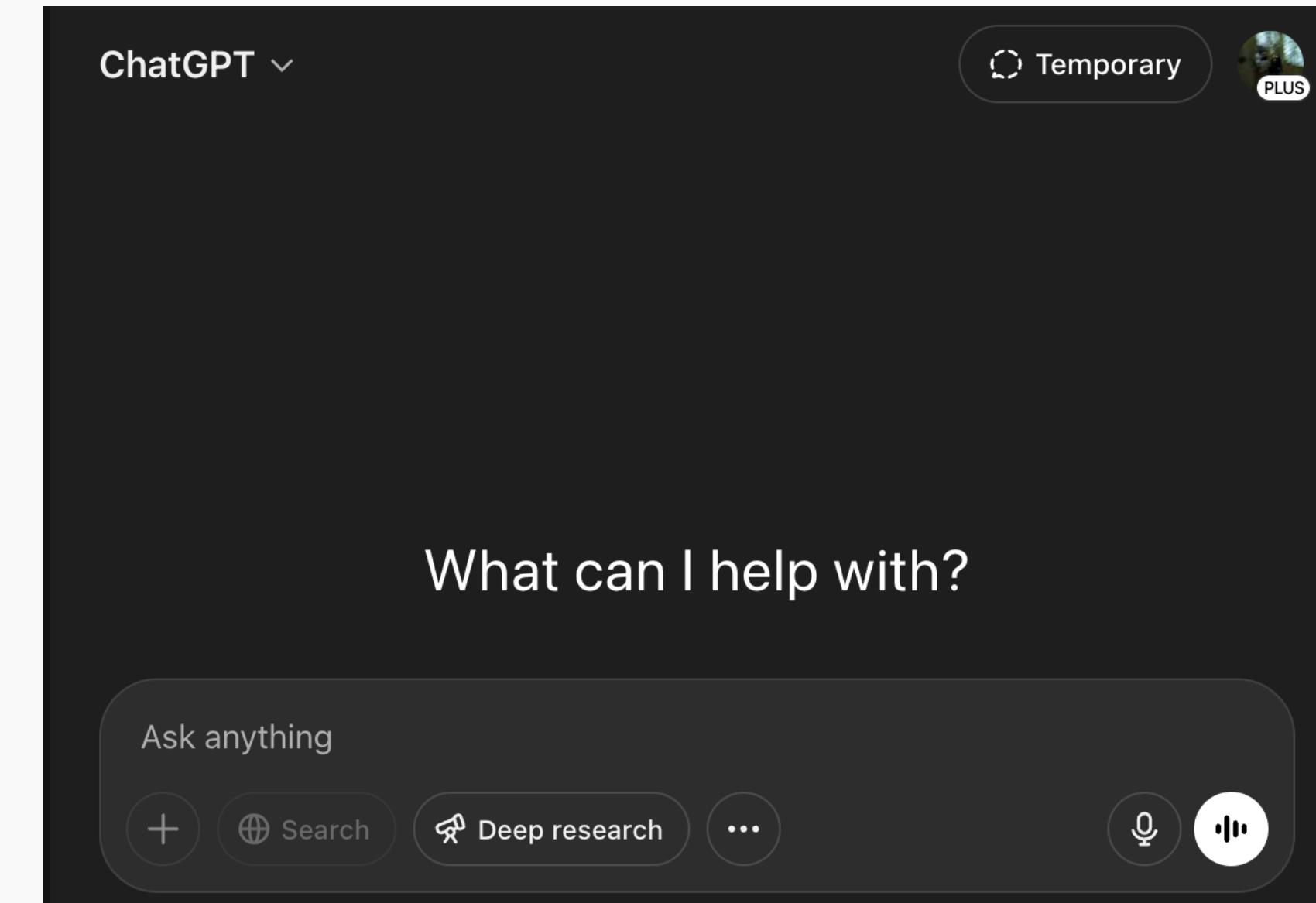
Carney, D. R., Cuddy, A. J., & Yap, A. J. (2010). Power posing: brief nonverbal displays affect neuroendocrine levels and risk tolerance. *Psychological Science*, 21(10), pp.1363-1368

...and many other purposes!

- generating “silicon samples” (Hewitt et al., 2025)
- Identifying “jingle-jangle” fallacies (Wulff & Mata, 2024)
- predicting correlations between scale items (Hommel & Arslan, 2025)
- identifying numeric errors in papers

How can we interact with LLMs?

Through a chat interface:



Programmatically, using an API connection:

```
blinded_response = openai_client.chat.completions.create(  
    model="gpt-4o-mini",  
    temperature=0,  
    messages=[{"role": "user", "content": blinded_prompt}  
])
```

Why would we use LLMs programmatically?

it's more flexible

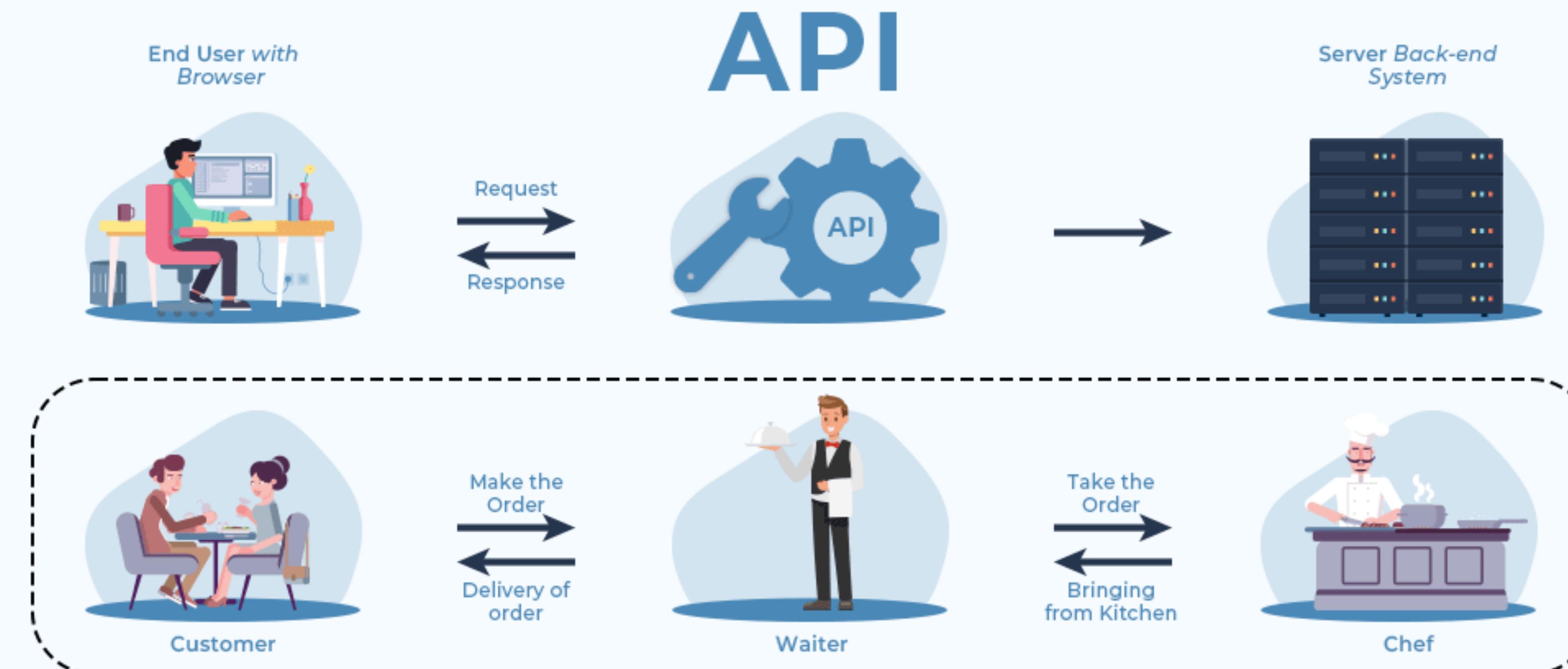
it's more reproducible

we control more parts of the workflow

it enables us to run many, many requests in quick succession

How do we use LLMs programmatically?

We use an application programming interface (API).



How do we use LLMs programmatically?

We use an application programming interface (API).

To keep track of requests for expenses and permissions,
we need to use an **API key**.

For today I will supply you all with a temporary API key.
Please don't abuse it because it costs money!

What about handling sensitive data?

...it kind of depends how much you trust OpenAI...

OpenAI's stated policy for API usage:

Security and data privacy

- ✓ No training on your data
- ✓ Zero data retention policy by request
- ✓ Business Associate Agreements (BAA)
for HIPAA compliance
- ✓ SOC 2 Type 2 compliance
- ✓ Single sign-on (SSO) and multi-factor authentication (MFA)
- ✓ Data encryption at rest (AES-256) and in transit (TLS 1.2+)
- ✓ Private Link to securely connect your Azure instances

We can use open source models instead.

Project	Availability						Documentation				Access			
	Open code	LLM data	LLM weights	RL data	RL weights	License	Code	Architecture	Preprint	Paper	Modelcard	Datasheet	Package	API
OLMo 7B Instruct	✓	✓	✓	✓	✓	✓	✓	✓	✓	X	✓	✓	✓	~
BLOOMZ	✓	✓	✓	✓	~	~	✓	✓	✓	✓	✓	✓	X	✓
AmberChat	✓	✓	✓	✓	✓	✓	~	~	✓	X	~	~	X	✓
Open Assistant	✓	✓	✓	✓	X	✓	✓	✓	~	X	X	X	✓	✓
OpenChat 3.5 7B	✓	X	✓	X	✓	✓	~	✓	✓	✓	~	X	✓	~
Pythia-Chat-Base-7...	✓	✓	✓	✓	X	✓	✓	✓	✓	~	X	~	✓	X
Cerebras GPT 111...	~	✓	✓	✓	✓	~	X	✓	~	X	X	✓	X	✓
RedPajama-INCITE-...	~	✓	✓	✓	✓	~	~	~	X	X	✓	✓	X	~
dolly	✓	✓	✓	✓	X	✓	✓	✓	~	X	X	X	✓	X
Tulu V2 DPO 70B	✓	X	~	✓	✓	~	~	~	✓	X	~	~	X	✓
MPT-30B Instruct	✓	~	✓	~	X	✓	✓	~	X	X	~	X	✓	~
MPT-7B Instruct	✓	~	✓	~	X	✓	✓	~	X	X	✓	X	✓	X
trlx	✓	✓	✓	~	X	✓	✓	~	X	X	X	X	~	✓
Vicuna 13B v 1.3	✓	~	✓	X	X	~	✓	X	✓	X	~	X	✓	~
minChatGPT	✓	✓	✓	~	X	✓	✓	~	X	X	X	X	X	✓
ChatRWKV	✓	~	✓	X	X	✓	~	~	~	X	X	X	✓	~
BELLE	✓	~	~	~	~	X	~	✓	✓	X	X	~	X	X
WizardLM 13B v1.2	~	X	~	✓	✓	~	~	✓	✓	X	X	X	X	X
Airoboros L2 70B G...	~	X	~	✓	✓	~	~	~	X	X	~	~	X	X
ChatGLM-6B	~	~	✓	X	X	✓	~	~	X	~	X	X	X	✓
Mistral 7B-Instruct	~	X	✓	X	~	✓	X	~	~	X	X	X	~	✓
WizardLM-7B	~	~	X	✓	~	~	~	✓	✓	X	X	X	X	X
Qwen 1.5	~	X	✓	X	✓	X	~	~	X	X	X	X	~	✓
StableVicuna-13B	~	X	~	~	~	~	~	~	~	X	~	X	X	~
Falcon-40B-instruct	X	~	✓	~	X	✓	X	~	~	X	~	X	X	X
UltraLM	X	X	~	✓	~	X	X	~	✓	X	~	~	X	X
Yi 34B Chat	~	X	✓	X	✓	~	X	X	✓	X	X	X	X	~
Koala 13B	✓	~	~	~	X	~	~	~	X	X	X	X	X	X
Mixtral 8x7B Instruct	X	X	✓	X	~	✓	X	~	~	X	X	~	X	X
Stable Beluga 2	X	X	~	X	✓	~	X	~	~	X	~	X	X	~
Stanford Alpaca	✓	X	~	~	~	X	~	✓	X	X	X	X	X	X
Falcon-180B-chat	X	~	~	~	~	X	X	~	~	X	~	X	X	X
Orca 2	X	X	~	X	✓	X	X	~	~	X	~	X	X	~
Command R+	X	X	X	✓	✓	~	X	X	X	X	~	X	X	X
Gemma 7B Instruct	~	X	~	X	~	X	X	~	X	X	✓	X	X	X
LLaMA2 Chat	X	X	~	X	~	X	X	~	~	X	~	X	X	~
Nanbeige2-Chat	✓	X	X	X	✓	~	X	X	X	X	X	X	X	~
Llama 3 Instruct	X	X	~	X	~	X	X	~	X	X	~	X	X	~
Solar 70B	X	X	~	X	~	X	X	X	X	X	~	X	X	~
Xwin-LM	X	X	~	X	X	X	X	X	X	X	X	X	X	~
ChatGPT	X	X	X	X	X	X	X	X	~	X	X	X	X	X

Liesenfeld et al. (2024). Rethinking open source generative AI: open-washing and the EU AI Act. *The ACM Conference on Fairness, Accountability, and Transparency.*

But there are practical challenges to going open source!

Like: how do we use the bigger ones?

Groq is Fast AI Inference

✖ Try it ➤



Hugging Face

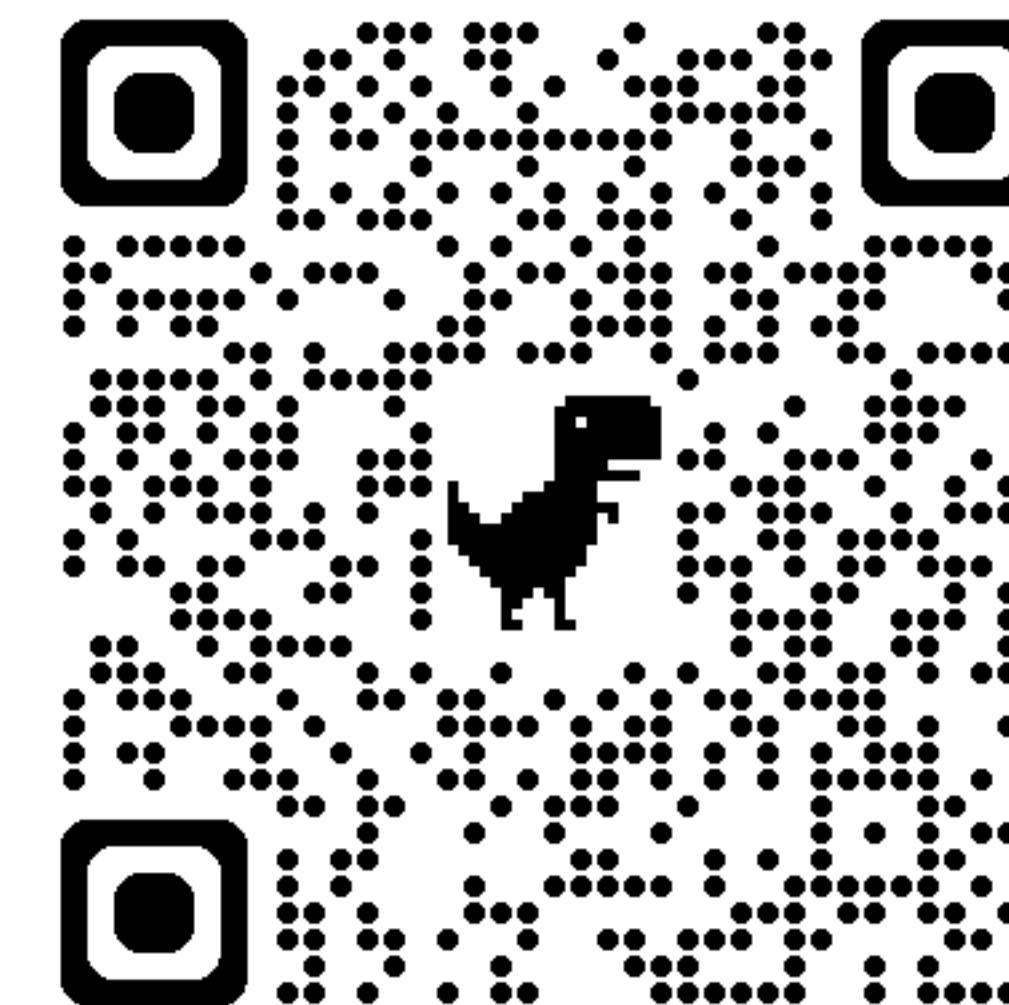
So why do people still use proprietary models?

Rank* (UB) ▲	Rank (StyleCtrl) ▲	Model ▲	Arena Score ▲	95% CI ▲	Votes ▲	Organization	License ▲
1	1	Gemini-2.5-Pro-Preview-05-06	1446	+8/-9	4500	Google	Proprietary
2	1	o3-2025-04-16	1413	+8/-7	6689	OpenAI	Proprietary
2	3	ChatGPT-4o-latest_(2025-03-26)	1408	+6/-6	10290	OpenAI	Proprietary
3	5	Grok-3-Preview-02-24	1403	+4/-4	14843	xAI	Proprietary
4	3	GPT-4.5-Preview	1398	+4/-5	15275	OpenAI	Proprietary
4	5	Gemini-2.5-Flash-Preview-04-17	1394	+7/-7	5959	Google	Proprietary
7	5	DeepSeek-V3-0324	1373	+7/-5	8753	DeepSeek	MIT
7	5	GPT-4.1-2025-04-14	1366	+7/-8	5102	OpenAI	Proprietary
8	8	DeepSeek-R1	1358	+4/-4	18493	DeepSeek	MIT
8	15	Gemini-2.0-Flash-001	1355	+4/-3	24913	Google	Proprietary
8	13	Hunyuan-Turbos-20250416	1355	+9/-10	3699	Tencent	Proprietary
8	5	o4-mini-2025-04-16	1351	+10/-7	5083	OpenAI	Proprietary
10	8	o1-2024-12-17	1350	+4/-4	29036	OpenAI	Proprietary
10	14	Qwen3-235B-A22B	1343	+11/-9	3611	Alibaba	Apache 2.0

Any questions?



Let's take a break and then play with some code!



[https://github.com/JamieCummins/
IIm-workshop-bern-2025](https://github.com/JamieCummins/IIm-workshop-bern-2025)