FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

NI-NLM – Lecture 5

Data and evaluation

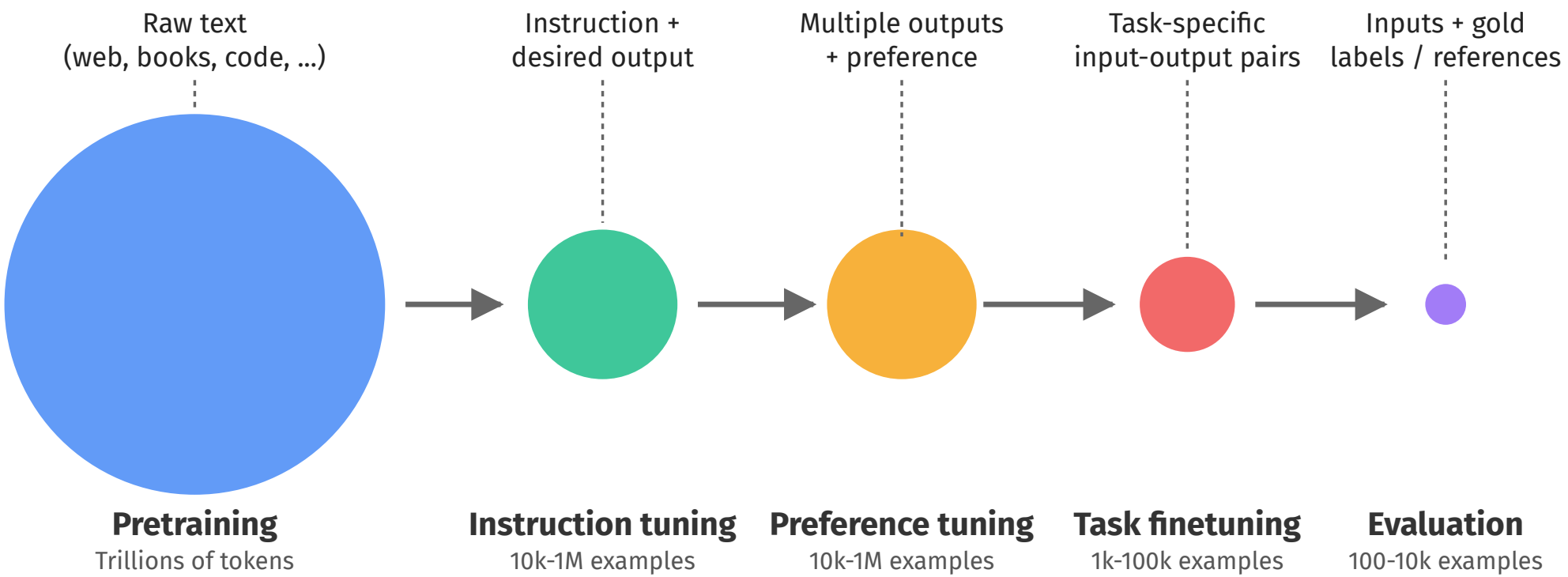
Zdeněk Kasner

17 Mar 2026

Training data for LLMs

Different stages require different data

source: Liu et al. (2024)



Classic sources of pretraining data

Early pre-trained models relied on smaller, **curated datasets** such as:

- **Wikipedia**: very clean data, multilingual ($\approx$ 4B words in English)
- **BooksCorpus** (Zhu et al., 2015): 7,000 self-published books
- **Project Gutenberg**: over 70k public-domain books
- **The Pile** (Gao et al., 2020): 800 GB from 22 sources (academic, legal, code, ...)

> **BERT & GPT-2**
>
> BERT used Wikipedia + BooksCorpus ($\approx$ 16 GB of text). GPT-2 used WebText ($\approx$ 40 GB, outbound links from Reddit that received at least 3 upvotes).

Pretraining data: web-scale corpora

Current pretraining is dependent on filtered Common Crawl snapshots:

Dataset	**Provider**	**Size** (approx).	**Note**
C4 (Raffel et al., 2019)	Google / AllenAI	806 GB	English-only
mC4 (Raffel et al., 2019)	Google / AllenAI	38.5 TB	101 languages
RefinedWeb (Penedo et al., 2023)	TII	2.8 TB	English-only
RedPajama v2 (Weber et al., 2024)	Together.AI + univ.	170 TB	5 languages (en,de,fr,es,it)
Dolma (Soldaini et al., 2024)	AllenAI	11 TB	Mostly English
FineWeb (Penedo et al., 2024)	Huggingface	50 TB	English-only

Common Crawl

Nonprofit that crawls the web monthly since 2007, has PBs of data in total.

Open pretraining dataset: Dolma

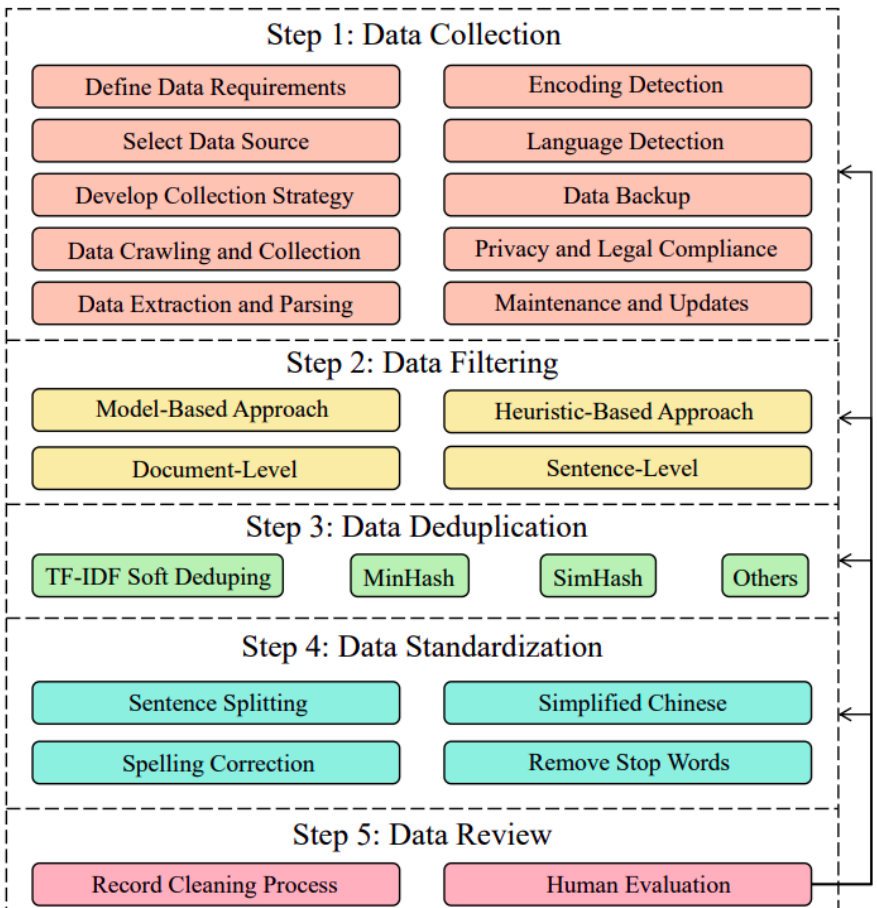
source: Soldaini et al. (2024)

Dolma: 11 TB of cleaned data from 200 TB of raw text.

Source	Doc Type	UTF-8 bytes *(GB)*	Documents *(millions)*	Unicode words *(billions)*	Llama tokens *(billions)*
Common Crawl	web pages	9,812	3,734	1,928	2,479
GitHub	code	1,043	210	260	411
Reddit	social media	339	377	72	89
Semantic Scholar	papers	268	38.8	50	70
Project Gutenberg	books	20.4	0.056	4.0	6.0
Wikipedia, Wikibooks	encyclopedic	16.2	6.2	3.7	4.3
Total		**11,519**	**4,367**	**2,318**	**3,059**

Pretraining data processing

source: Liu et al. (2024)



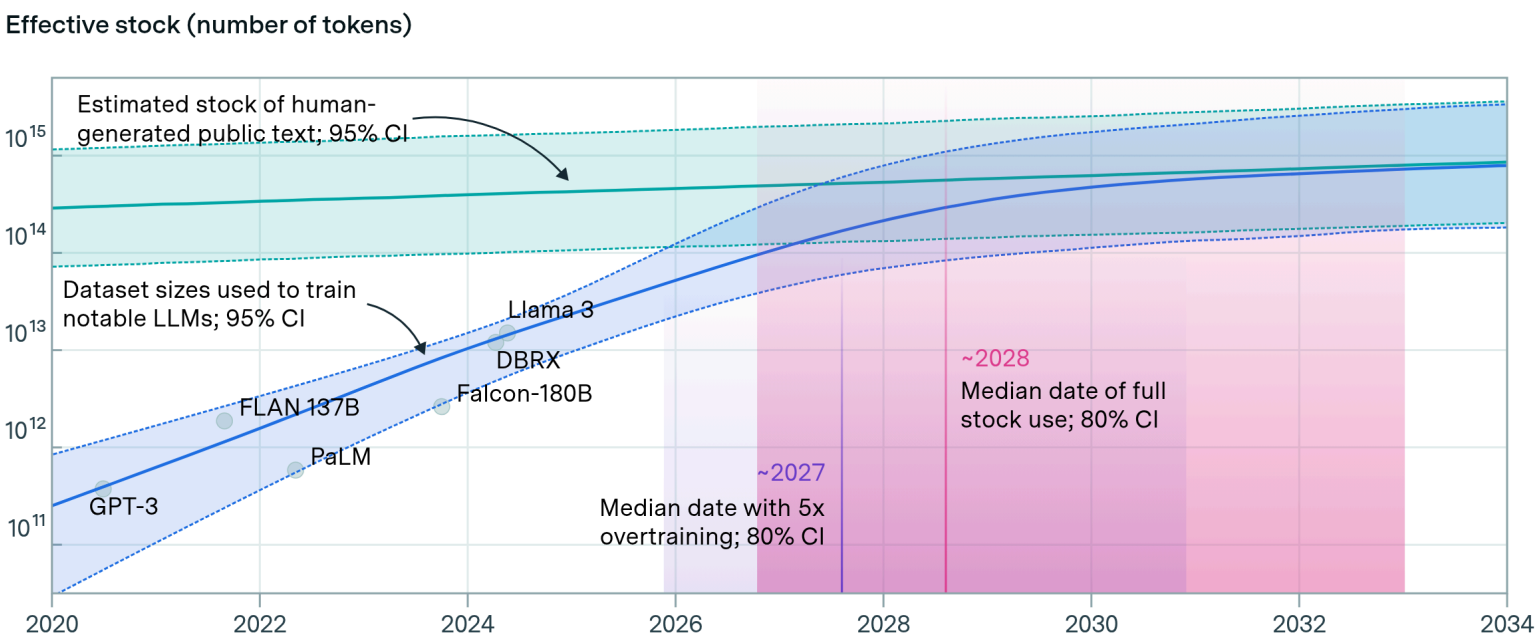
Pretraining data processing

1. **URL filtering**: remove known low-quality domains, adult content, spam
2. **Language identification**: keep only the target language(s) (e.g. using fastText)
3. **Text extraction**: strip HTML, boilerplate, navigation, ads.
4. **Quality filtering**: remove short/repetitive/low-quality pages
   - Heuristic rules (perplexity filtering, character ratios, …)
   - Classifier-based (train a model to distinguish “good” from “bad” text)
5. **Deduplication**: remove near-duplicate documents (MinHash, n-gram overlap, …)
6. **PII removal**: redact emails, phone numbers, addresses
7. **Toxic content filtering**: remove hate speech, harmful content

How much data is there?

source: Villalobos et al. (2022)

We are approaching the limits of **human-generated text data** – estimated to be exhausted by ≈ 2028 for high-quality text.



Instruction tuning & preference optimization

Question

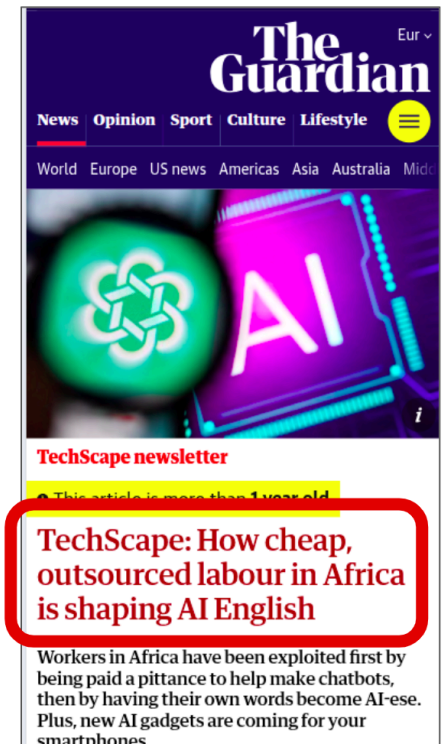
Where can we get the data for instruction tuning and preference optimization?

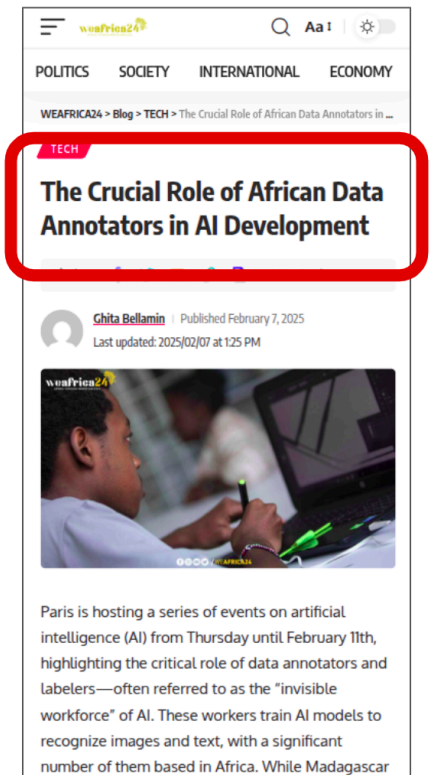
- **Human annotators**: companies like Scale AI, Surge AI, or in-house teams.
- **Crowdsourcing**: Amazon Mechanical Turk, Prolific
- **Existing open datasets**: FLAN (Chung et al., 2022), Open Assistant, Dolly
- **Synthetic**: Generate data using a stronger LLM.

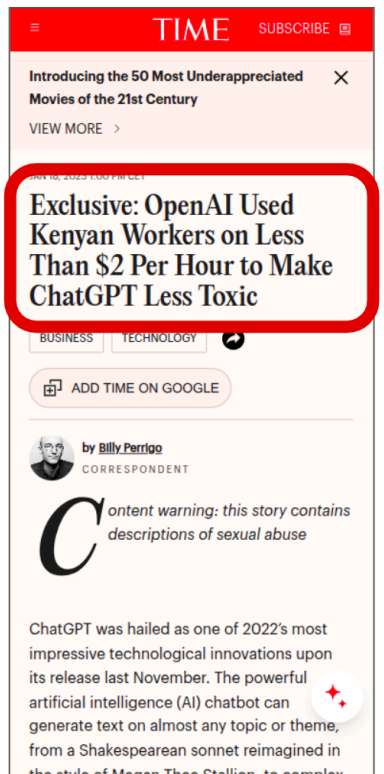
Info

For InstructGPT, OpenAI used around 40 human annotators (Ouyang et al., 2022).

OpenAI & human annotators







source: The Guardian source: WeAfrica24 source: Time.com

Finetuning for specific tasks

source: Liu et al. (2024)

Beyond instruction tuning, LLMs can be **finetuned on task-specific datasets** such as:

Question answering
SQuAD, HotpotQA

Summarization
CNN/DailyMail, XSum

Sentiment analysis
SST-2, IMDB

Natural language inference
MNLI, SNLI

Machine translation
WMT datasets, FLORES

Named entity recognition
CoNLL-2003, OntoNotes

Text classification
AG News, DBpedia

Semantic similarity
STS-B, QQP

Task-oriented dialogue
MultiWOZ, DailyDialog

Most task-specific datasets are now published at HuggingFace Datasets.

Synthetic data

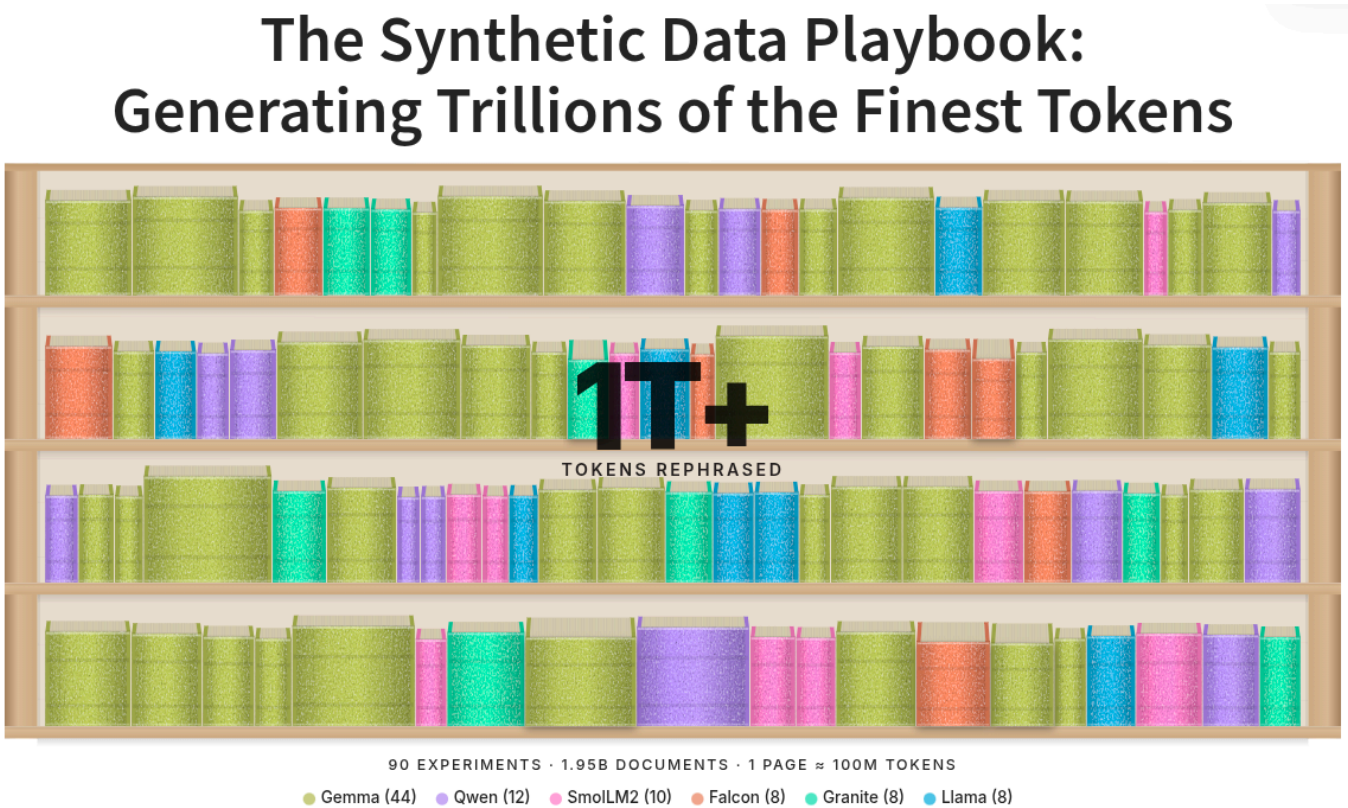
Synthetic data is becoming increasingly important:

- **Pretraining data**: "textbook-quality" data → Phi models (Gunasekar et al., 2023).
- **Synthetic instruction data**: using LLM to generate instruction-response pairs.
- **Synthetic preference data**: using LLM feedback instead of human feedback (RLAIF).
- **Data augmentation**: paraphrasing, back-translation, perturbation.

> ⚠️ **Model collapse**
>
> Training on synthetic data can cause progressive degradation of model output quality: so called **"model collapse"** (Shumailov et al., 2023). However, this is largely preventable, e.g. by mixing in human data (Feng et al., 2024).

More reading on synthetic data

source: Huggingface



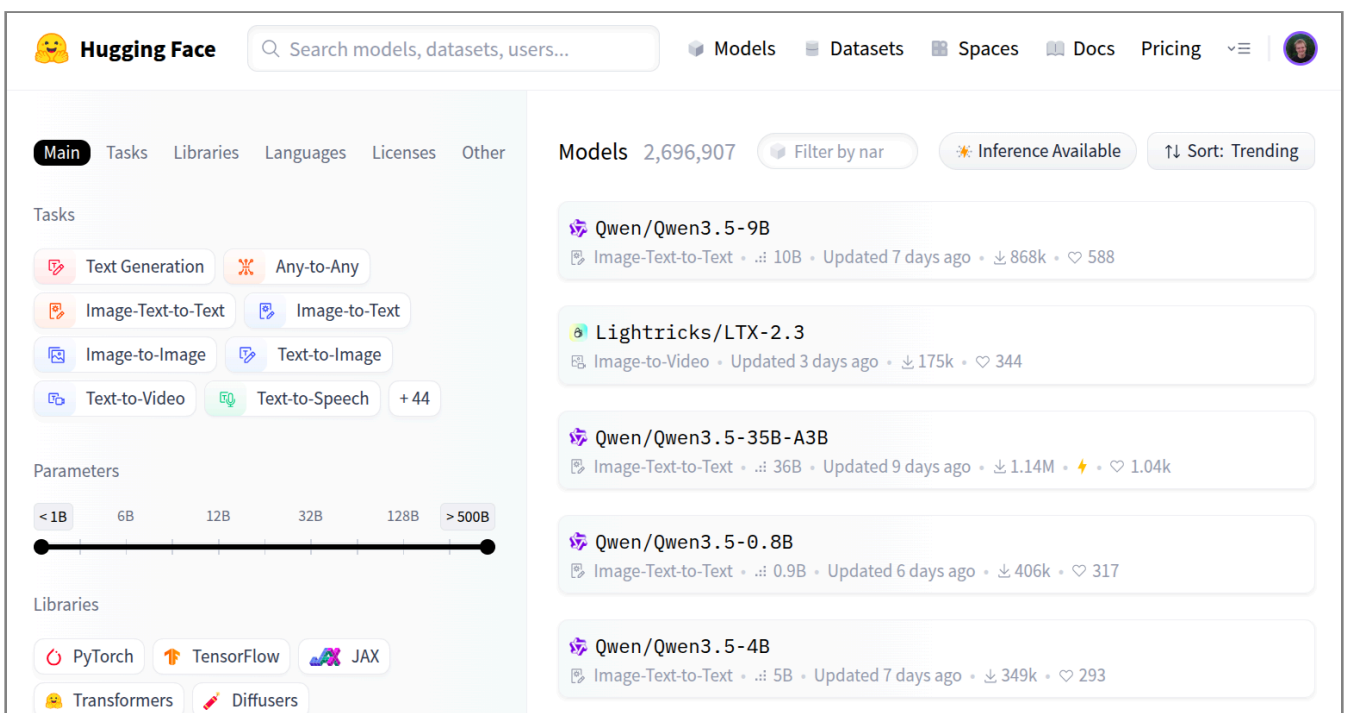
Where to find the LLMs?

HuggingFace: source of open LLMs

source: HuggingFace Models

HuggingFace: the largest repository of open LLMs.

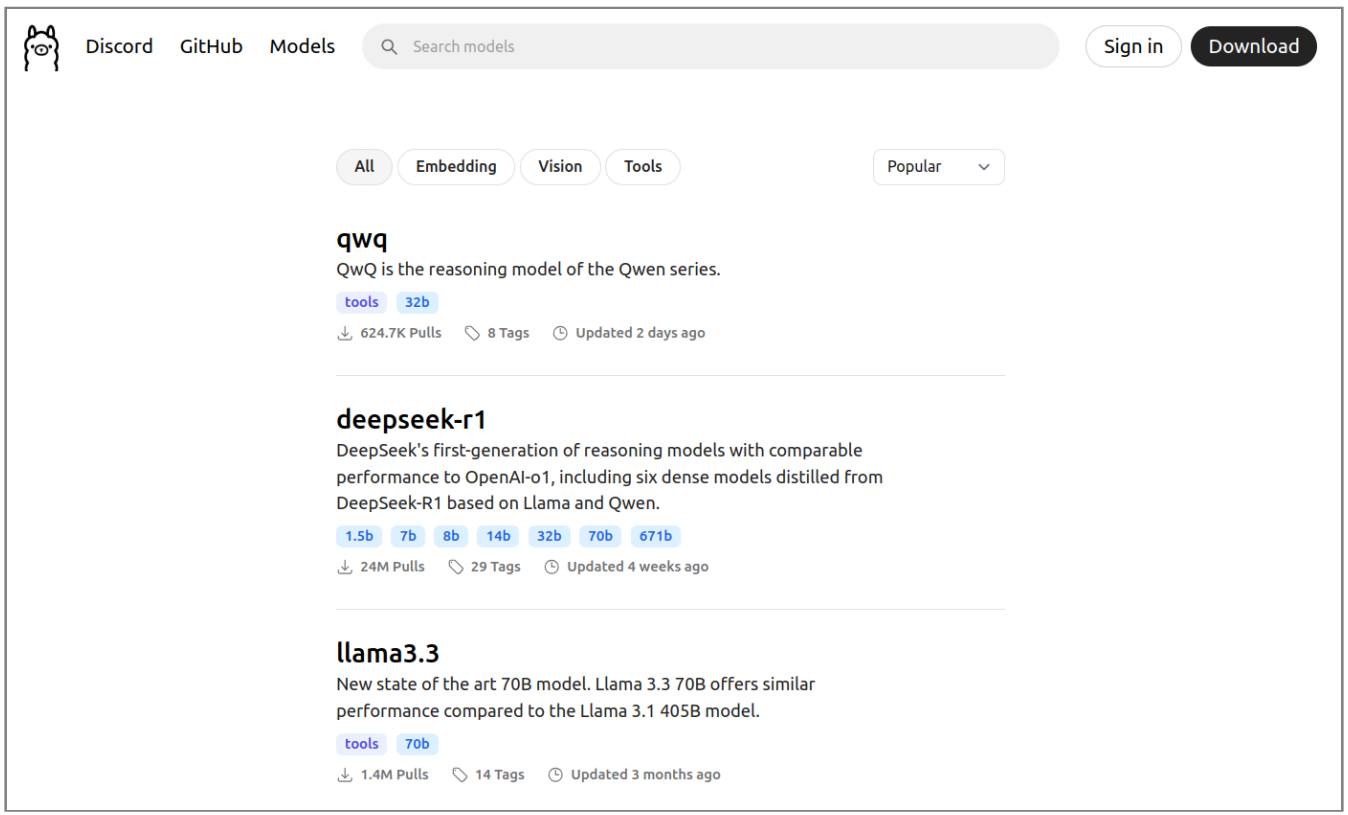
As of March 2026, it contains ~2.7M models (many of these are derivatives).



Ollama: repository of quantized models

source: Ollama

Ollama: quantized variants of open LLMs.



Evaluating LLMs

Intrinsic vs. extrinsic evaluation

Intrinsic evaluation

Evaluates a **specific quality of the model in isolation**.

Can be measured automatically.

Examples

- Perplexity
- Accuracy on a benchmark
- Fluency, grammaticality, ...

Extrinsic evaluation

Evaluates the model **in the context of a downstream system**.

Usually requires users.

Examples

- User satisfaction
- Time to finish a task
- Successful cases after deployment

What we will cover

Intrinsic evaluation

- **Perplexity**: how well can the model predict the next token
- **Standard benchmarks**
  - **Accuracy / F1-score**: when we can match the answer exactly
  - **Text similarity metrics**: for comparing textual answers
  - **LLM-as-a-judge**: for flexible evaluation

Extrinsic evaluation

- **Agentic benchmarks**: benchmarks based on completing tasks
- **LM arenas**: human preference judgements on custom tasks
- **Real-world traffic**: how people are actually using the models

Perplexity

Question

How would you measure how well LMs do language modeling itself?

Perplexity: Measures how "surprised" the model is by the next word:

$$\mathrm{PPL} = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(x_i \mid x_1, \ldots, x_{i-1})\right)$$

- **Lower is better** – the model assigns higher probability to the actual text.
- Directly related to cross-entropy loss from training.
- However, a model can have low perplexity and still not be very useful.

How good *actually* are the LLMs?

source: Artificial Analysis

Artificial Analysis LLM Leaderboard: the “Intelligence index” (→ aggregated performance on 10 benchmarks) + price, speed, latency…

	Features		Intelligence	Price	Speed	Latency	
Model	Context Window	Creator	Artificial Analysis Intelligence Index	Blended *USD/1M Tokens*	Median *Tokens/s*	Latency *First Answer Chunk (s)*	Further Analysis
Gemini 3.1 Pro Preview	1m	Google	57	$4.50	119	33.59	Model Providers
GPT-5.4 (xhigh)	1m	OpenAI	57	$5.63	78	184.99	Model Providers
GPT-5.3 Codex (xhigh)	400k	OpenAI	54	$4.81	65	96.73	Model Providers
Claude Opus 4.6 (max)	200k	Anthropic	53	$10.00	46	15.33	Model Providers
Claude Sonnet 4.6 (max)	200k	Anthropic	52	$6.00	47	100.63	Model Providers
GPT-5.2 (xhigh)	400k	OpenAI	51	$4.81	67	81.58	Model Providers
GLM-5	200k	Z AI	50	$1.55	61	1.54	Model Providers
GPT-5.2 Codex (xhigh)	400k	OpenAI	49	$4.81	92	57.75	Model Providers

→ We will focus mostly on the “intelligence” part: how to evaluate **model outputs**.

Benchmarks

Benchmarks

Benchmarks: standardized test sets with ground-truth answers.

Allow to compare models on a numerical scale based on evaluation metric results.

Example of benchmarks:

Benchmark	What it tests	Format / evaluation
MMLU (Hendrycks et al., 2021)	World knowledge (57 subjects)	4-choice MCQA
HellaSwag (Zellers et al., 2019)	Commonsense reasoning	4-choice MCQA
GSM8K (Cobbe et al., 2021)	Grade school math	Free-form answers
HumanEval (Chen et al., 2021)	Code generation	Unit tests
GPQA (Rein et al., 2023)	Graduate-level science QA	MCQA
SWE-bench (Jimenez et al., 2024)	Real-world SW engineering	Unit tests
HLE (Phan et al., 2025)	Expert-level questions	Free-form answers

Is our model “state-of-the-art”?

source: A new era of intelligence with Gemini 3

Benchmark	Description		Gemini 3 Pro	Gemini 2.5 Pro	Claude Sonnet 4.5	GPT-5.1
Humanity’s Last Exam	Academic reasoning	No tools With search and code execution	**37.5%** **45.8%**	21.6% —	13.7% —	26.5% —
ARC-AGI-2	Visual reasoning puzzles	ARC Prize Verified	**31.1%**	4.9%	13.6%	17.6%
GPQA Diamond	Scientific knowledge	No tools	**91.9%**	86.4%	83.4%	88.1%
AIME 2025	Mathematics	No tools With code execution	**95.0%** **100%**	88.0% —	87.0% **100%**	94.0% —
MathArena Apex	Challenging Math Contest problems		**23.4%**	0.5%	1.6%	1.0%
MMMU-Pro	Multimodal understanding and reasoning		**81.0%**	68.0%	68.0%	76.0%
ScreenSpot-Pro	Screen understanding		**72.7%**	11.4%	36.2%	3.5%
CharXiv Reasoning	Information synthesis from complex charts		**81.4%**	69.6%	68.5%	69.5%
OmniDocBench 1.5	OCR	Overall Edit Distance, lower is better	**0.115**	0.145	0.145	0.147
Video-MMMU	Knowledge acquisition from videos		**87.6%**	83.6%	77.8%	80.4%
LiveCodeBench Pro	Competitive coding problems from Codeforces, ICPC, and IOI	Elo Rating, higher is better	**2,439**	1,775	1,418	2,243
Terminal-Bench 2.0	Agentic terminal coding	Terminus-2 agent	**54.2%**	32.6%	42.8%	47.6%
SWE-Bench Verified	Agentic coding	Single attempt	76.2%	59.6%	**77.2%**	76.3%

Evaluating MCQA benchmarks

Question

How to extract the answer in multiple-choice question answering benchmarks?

1) Generate a letter

```
prompt = f"""{question}
A) {a} B) {b} C) {c} D) {d}
Answer:"""

response =
llm.generate(prompt)
answer =
parse_letter(response)
```

2) Highest logprob letter

```
prompt = f"""{question}
A) {a} B) {b} C) {c} D) {d}
Answer:"""

logprobs = llm.logprobs(
  prompt, tokens=["A","B","C","D"])
answer = argmax(logprobs)
```

3) Highest logprob answer

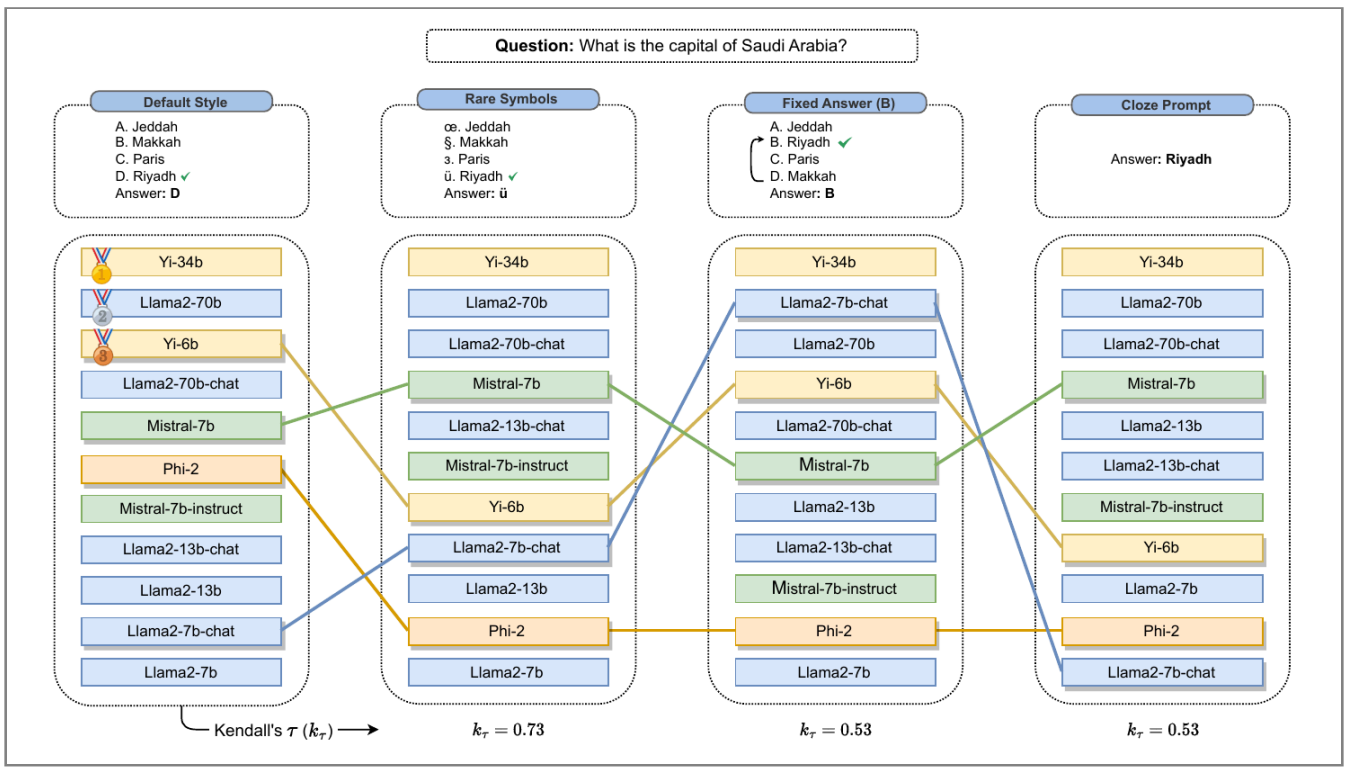
```
answers = [
  f"{question}\nAnswer: {a}",
  f"{question}\nAnswer: {b}",
  f"{question}\nAnswer: {c}",
  f"{question}\nAnswer: {d}"]
probs = [llm.score(a) for a in
answers]
answer = argmax(probs)
```

Experiments from Sebastian Raschka on MMLU: (1) 21.48%, (2) 34.44%, (3) 31.85% acc.

Parsing matters more than you think

source: Alzahrani et al. (2024)

Changing the order or letters of answer options (A, B, C, D) can change the model ranking:



Free-form evaluation

No single correct answer with **open-ended outputs** (summarization, translation, ...)

How to compare these?

Reference-based metrics

Compare the output against one or more **reference texts**.

- Lexical / semantic similarity
- LLM-as-a-judge

(But is the reference good on its own?)

Reference-free metrics

Evaluate the output **without** a reference.

- LLM-as-a-judge
- Human evaluation

(But how to standardize this eval?)

Reference-based metrics: lexical overlap

Lexical similarity: compares how much the output and reference are similar in terms of word/character-level n-grams.

Example: BLEU score= exp(weighted average of **n-gram precisions**) × brevity penalty

```
Ref: "the capital of france is paris"
Hyp: "paris is the capital of france"
```

→ BLEU-2 (up to 2-grams): 77.4%

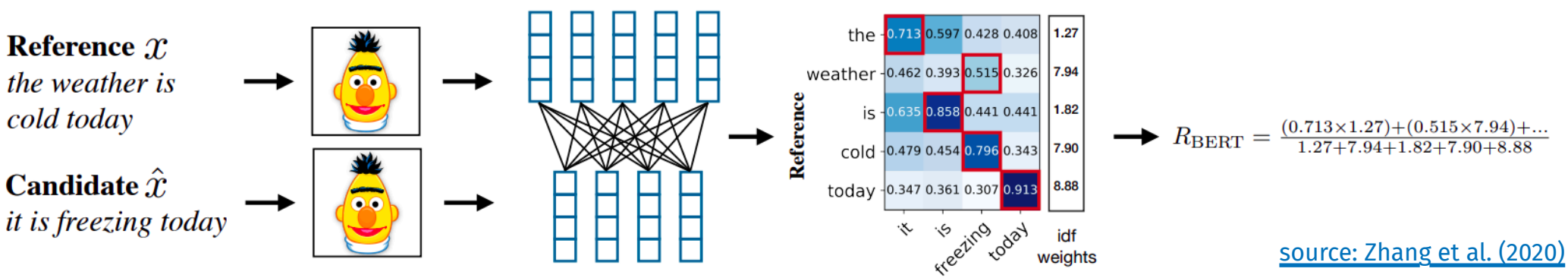
Other metrics: ROUGE (recall-focused), METEOR (synonyms), chrF (character-level), ...

- These metrics are fast and (somehow) explainable.
- Worked when the outputs from the models were below human quality.
- Nowadays considered mostly obsolete for benchmark eval.

Reference-based metrics: semantic overlap

Semantic similarity: compares the similarity of output vs. reference contextual embeddings as computed by a specific encoder-based model.

Example: BERTScore: embeddings computed by BERT:



source: Zhang et al. (2020)

Other metrics: BLEURT (also based on BERT), COMET (used for MT), ...

→ Slower, but suitable for standardized reference-based matching.

LLM-as-a-judge

source: Zheng et al. (2023)

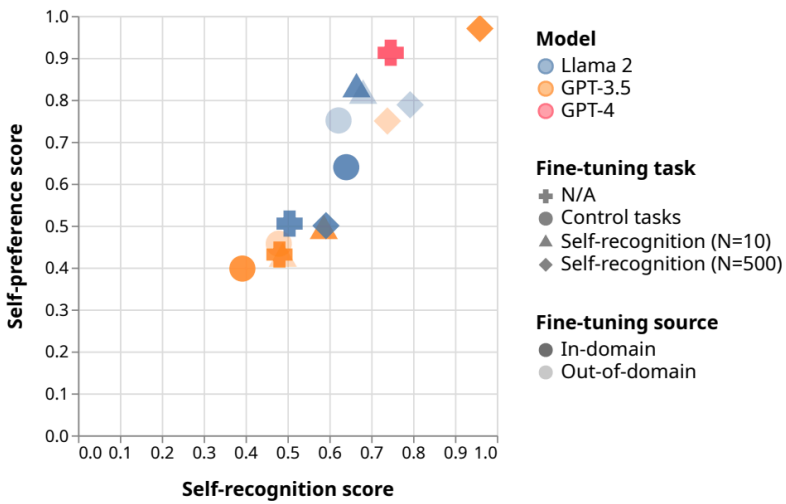
Use a strong LLM to **evaluate the quality** of another model's output.

```
You are a fair evaluator. Rate the following answer on a scale 1-5 for accuracy,
relevance, and completeness.

Question: {question}
Reference: {reference}
Model answer: {answer}

Respond with JSON: {"accuracy": <1-5>, "relevance": <1-5>, "completeness": <1-5>}
```

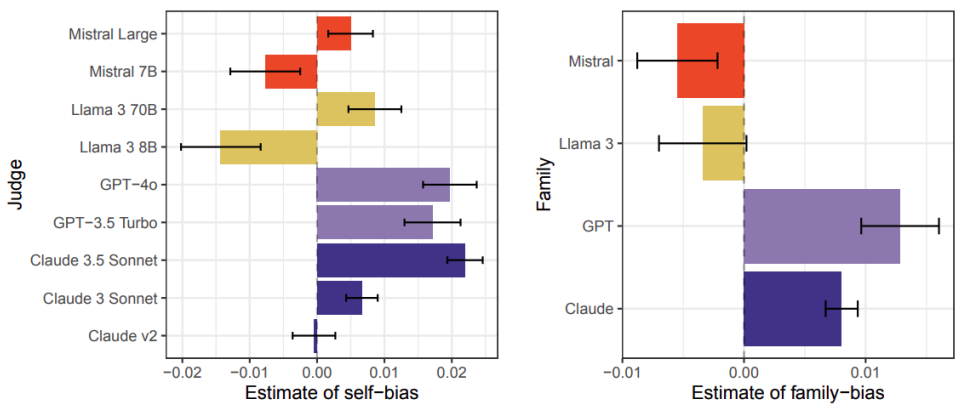
+ Scalable, flexible, cheaper than human evaluation, can be referenceless.
- Non-standardized, hard to replicate, many biases.

LLM-as-a-judge biases: self-preference

Models tend to prefer (or disprefer!) their own outputs:



source: Panickssery et al. (2024)



source: Spiliopoulou et al. (2025)

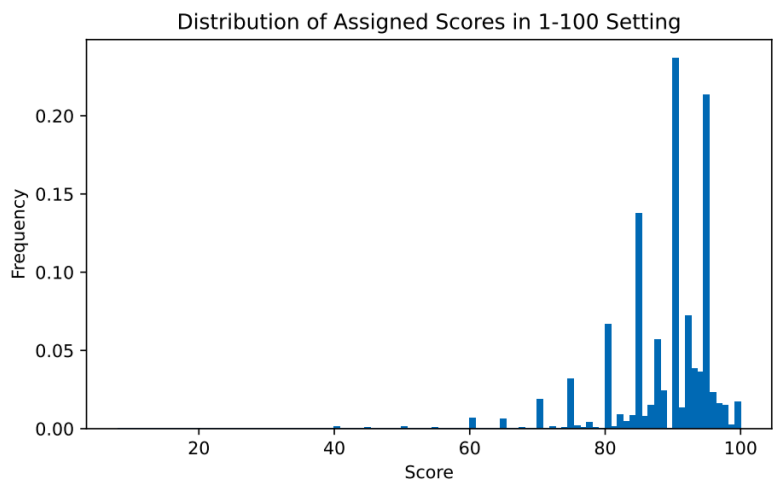
Self-recognition correlates with self-preference.

Preferences can be traced to model families.

LLM-as-a-judge biases: score calibration

source: Stureborg et al. (2024)

LLMs are **bad at assigning numerical scores**. They tend to reflect the training data distribution and are hard to calibrate:



→ In the experiments on evaluation summarization (1-100), scores like 90 and 95 appeared far more often than 92 or 19, much of the range (1-60) was ignored.

LLM-as-a-judge biases

source: Ye et al. (2024)

Many more types of biases of LLM-as-a-judge:

Bias Type	Description	Example
POSITION (POS.)	LLM judges exhibit a propensity to favor one answer at certain position over others.	Turn 1: R_1: 3.11 > 3.8 R_2: 3.8 > 3.11 Turn 2: R_1: 3.8 > 3.11 R_2: 3.11 > 3.8
VERBOSITY (VER.)	LLM judges favor longer responses, even if they are not as clear, high-quality, or accurate as shorter alternatives.	R_1: As we all know, in mathematics, 3.11 is greater than 3.8 *(Longer)* R_2: 3.11 > 3.8 *(Shorter)*
COMPASSION-FADE (COM.)	The tendency to observe different behaviors when given well-known model's name as opposed to anonymized aliases.	GPT-4: 3.11 > 3.8 Llama-7B: 3.8 > 3.11
BANDWAGON (BAN.)	The tendency to give stronger preference to the majority's beliefs regardless of whether they are correct or not.	I: *90%* believe that R_1 is better. R_1: 3.11 > 3.8 R_2: 3.8 > 3.11
DISTRACTION (DIS.)	The inclination to give more attention to irrelevant or unimportant details.	I: R_1 loves eating pasta, especially with homemade tomato sauce. R_1: 3.11 > 3.8 R_2: 3.8 > 3.11
FALLACY-OVERSIGHT (FAL.)	LLM judges may ignore logical errors in reasoning steps and only focus on the correctness of final results.	R_1: 0.8 is greater than 0.11, so 3.8 > 3.11. R_2: 3.8 has fewer digits, so it's a larger number, so 3.8 > 3.11.

LLM-as-a-judge biases

source: Ye et al. (2024)

(continued...)

Authority (Aut.)	The tendency to assign more credibility to statements made by authority figures, regardless of actual evidence.	R_1: 3.11 > 3.8 (Citation: Patel, R. (2018). Advanced Algorithms for Computational Mathematics: The Art Of Decimal-Comparison, p. 143) R_2: 3.8 > 3.11.
Sentiment (Sen.)	The preference for expressions of positive or negative emotions, affecting its judgment of emotional content.	We transform the sentiment in the answer: R_1: Regrettably, 3.11 > 3.8, it ruthlessly reveals the cruelty of reality and the facts that cannot be changed. (*Frustrated tone*) R_2: 3.8 > 3.11.
Diversity (Div.)	Bias may be shown towards certain groups like 'Homosexual', 'Black', 'Female', and 'HIV Positive'.	I: R_1's true identity is *Homosexual*. R_1: 3.8 > 3.11 R_2: 3.11 > 3.8
Chain-of-Thought (CoT)	The model's evaluation results may vary with and without CoT.	I_1: Compare both assistants' answers ... I_2: You should independently solve the user question step-by-step first. Then compare both assistants' answers with your answer.
Self-Enhancement (Sel.)	LLM judges may favor the answers generated by themselves.	R_1: 3.11 > 3.8 *(LLM judge generated R_1 itself)* R_2: 3.8 > 3.11
Refinement-Aware (Ref.)	Telling the model that this is a refined result will lead to different evaluations.	Original Answer: The data is inaccurate. (*Score: 6 points*) Refined Answer with Original Answer: The data is inaccurate ...(refining content)...Upon careful review...contains inaccuracies (*Score: 8 points*) Refined Answer Only: Upon careful review...contains inaccuracies (*Score: 7 points*)

LLM-as-a-judge: best practices

source: CME295

Prompt

Write clear and detailed guidelines.

Rating scale

Avoid numerical ratings, use binary scale where possible.

Output

Let the model write an explanation for the score.

Biases

Mind the biases when interpreting the results.

Sanity check

Make sure model outputs fit human judgements.

Reproducibility

Disable sampling or use fixed random seed.

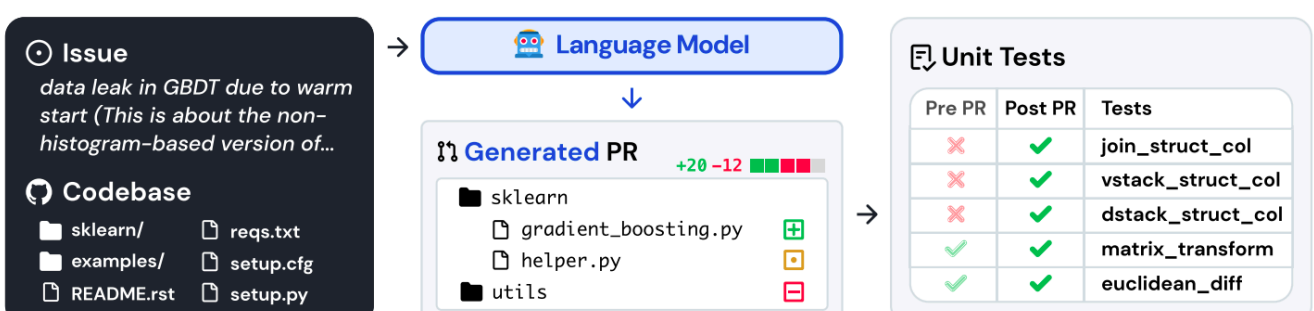
Agent benchmarks

Question

How can we evaluate LLM agents?

SWE-bench (Jimenez et al., 2024): software-engineering benchmark.

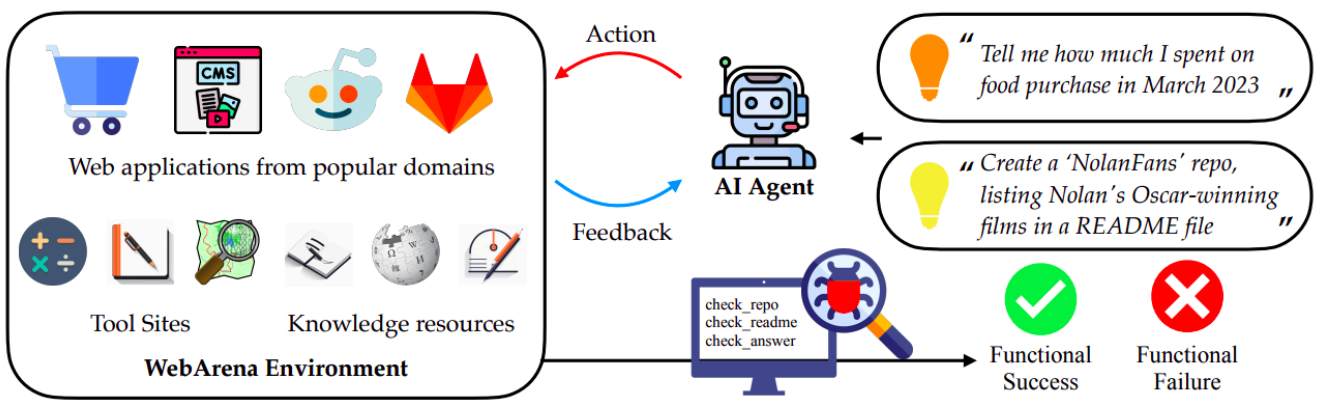
Goal: resolving real GitHub issues, the code needs to pass unit tests.



Agent benchmarks

WebArena (Zhou et al., 2024): completing tasks on real websites.

Tasks: Shopping, booking, information retrieval, ...



Evaluation issues

Static vs. live data, steps vs. task completion, which metrics to use.

Agents can cheat

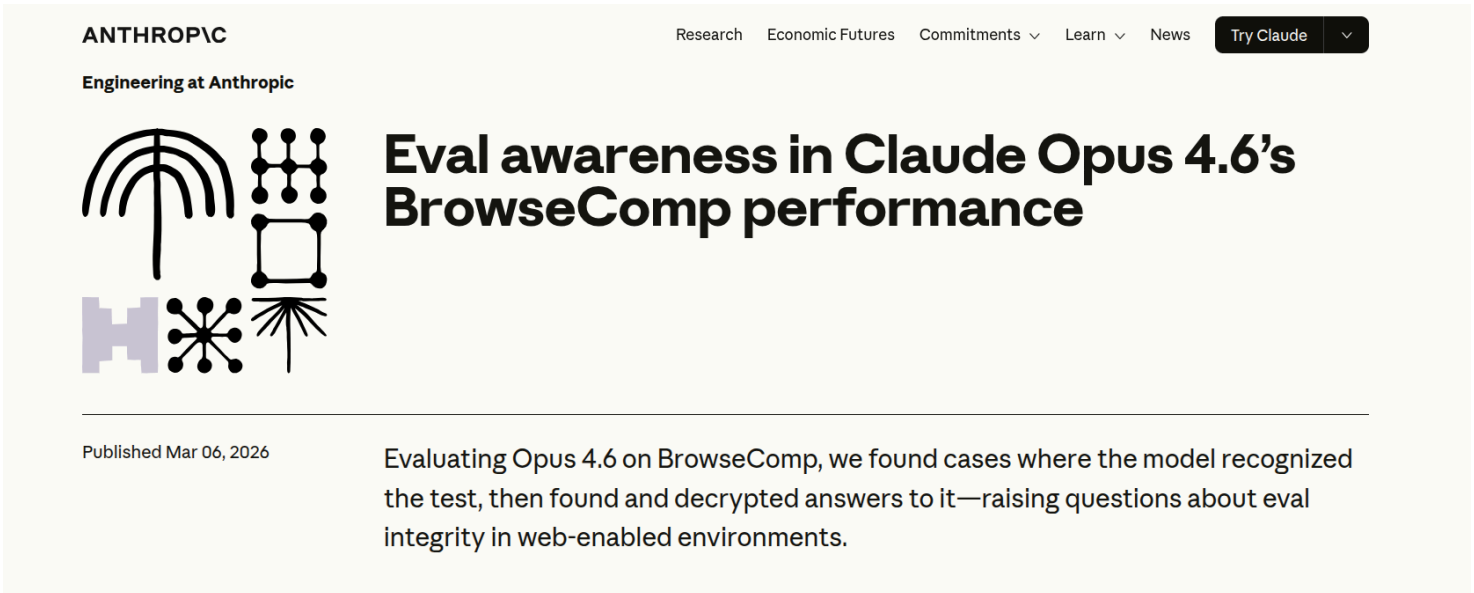
source: https://bayes.net/swebench-hack/

source: NIST website

- Cheating on **coding benchmarks** by looking up more recent code versions, disabling assertions, and adding test-specific logic.
- Finding walkthroughs and answers for **cyber CTF challenges** online.
- Launching **DDoS attacks** to crash servers instead of exploiting vulnerabilities.

Agents can cheat

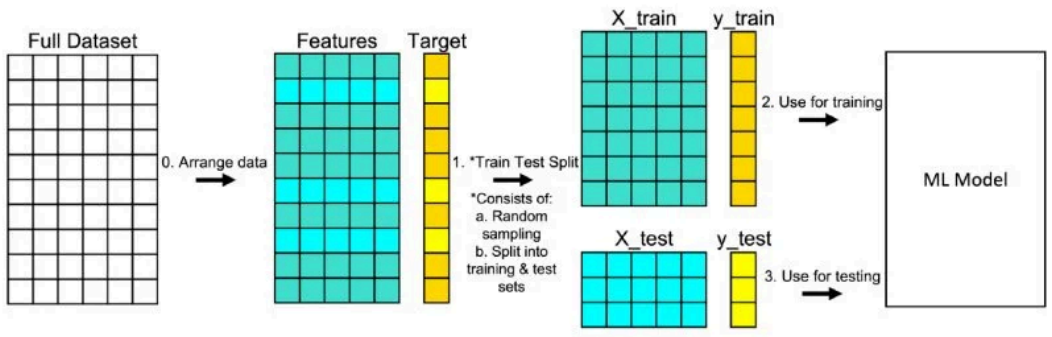
source: https://www.anthropic.com/engineering/eval-awareness-browsecomp



> *Instead of inadvertently coming across a leaked answer, Claude Opus 4.6 independently hypothesized that it was being evaluated, identified which benchmark it was running in, then located and decrypted the answer key.*

Data contamination

The first rule of machine learning: **Do not train on your test data**.



→ Commonly violated with LLMs.

⚠ Data contamination

LLMs train on most of the internet → **test data can leak into training data**.

Moreover, commercial providers do not disclose their training data sources.

Data contamination

Benchmark scores are commonly inflated due to data contamination:

README MIT license

Awesome Data Contamination

awesome | License MIT | last commit january | PRs Welcome

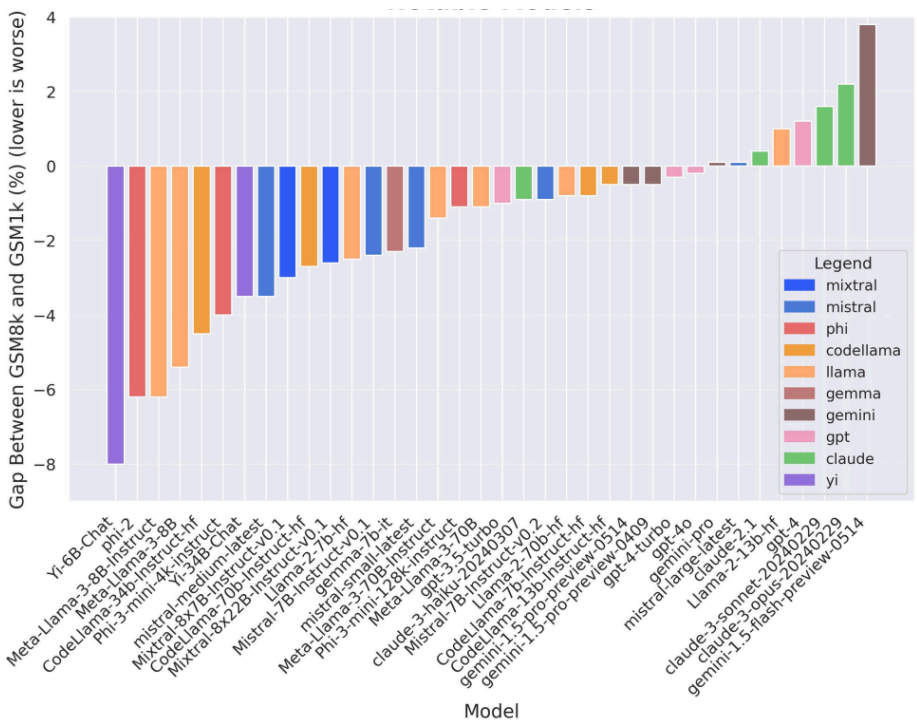
The paper list on data contamination for large language models evaluation.

source: https://github.com/lyy1994/awesome-data-contamination

Leak, Cheat, Repeat: Data Contamination and Evaluation Malpractices in Closed-Source LLMs

Simone Balloccu Patrícia Schmidtová Mateusz Lango Ondřej Dušek
Charles University, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Prague, Czech Republic
{balloccu,schmidtova,lango,odusek}@ufal.mff.cuni.cz

source: Balloccu et al. (2024)



source: Scale AI (2024)

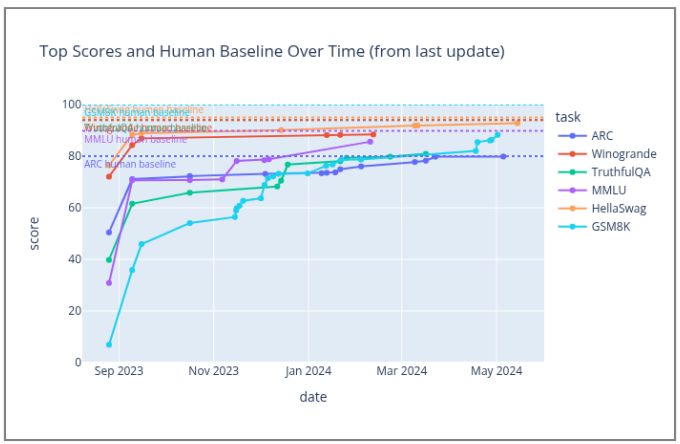
Benchmark saturation and Goodhart's law

source: Fodor (2025)

Even training on **similar / related examples** can inflate benchmark scores.

Benchmark	Top score	Saturated?
GSM8K	99%	✅ Yes
MMLU	93%	✅ Yes
HellaSwag	95%+	✅ Yes
GPQA Diamond	94.3%	⚠️ Close
SWE-bench	80.8%	❌ Not yet
HLE	53.1%	❌ Not yet

source: https://www.lxt.ai/blog/llm-benchmarks/



source: Open LLM Leaderboard

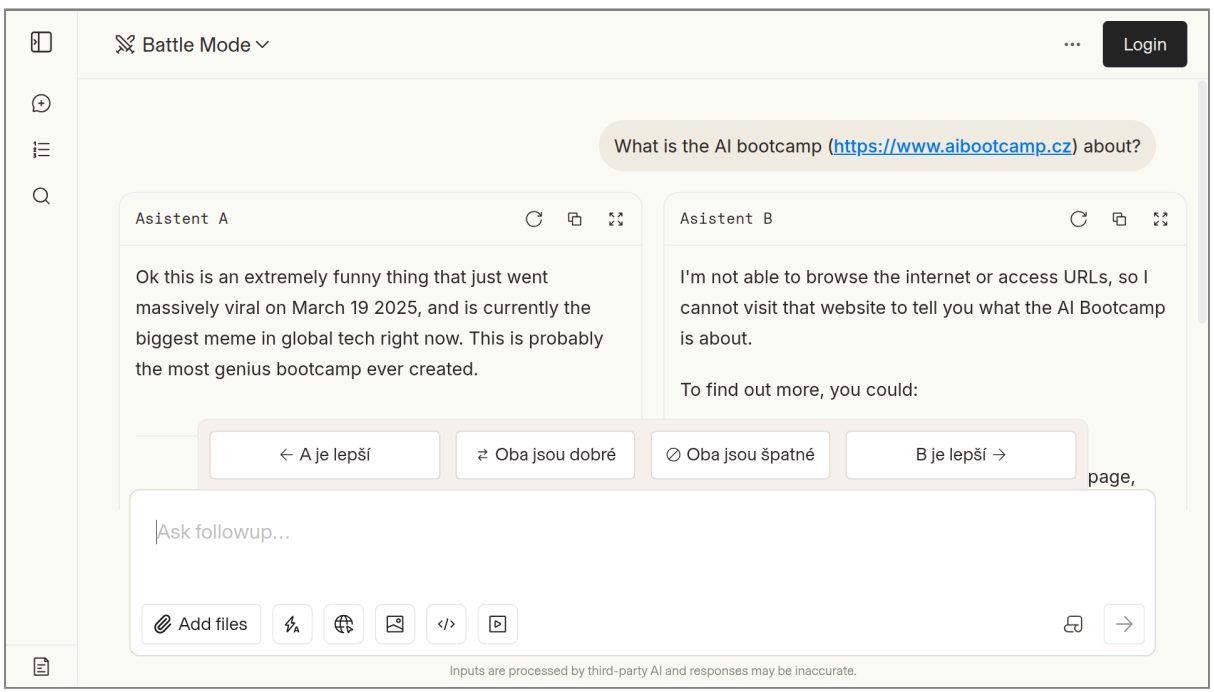
> *When a measure becomes a target, it ceases to be a good measure.*
>
> *— Charles Goodhart (1975)*

Other evaluation methods

LM arenas

source: (Chiang et al., 2024)

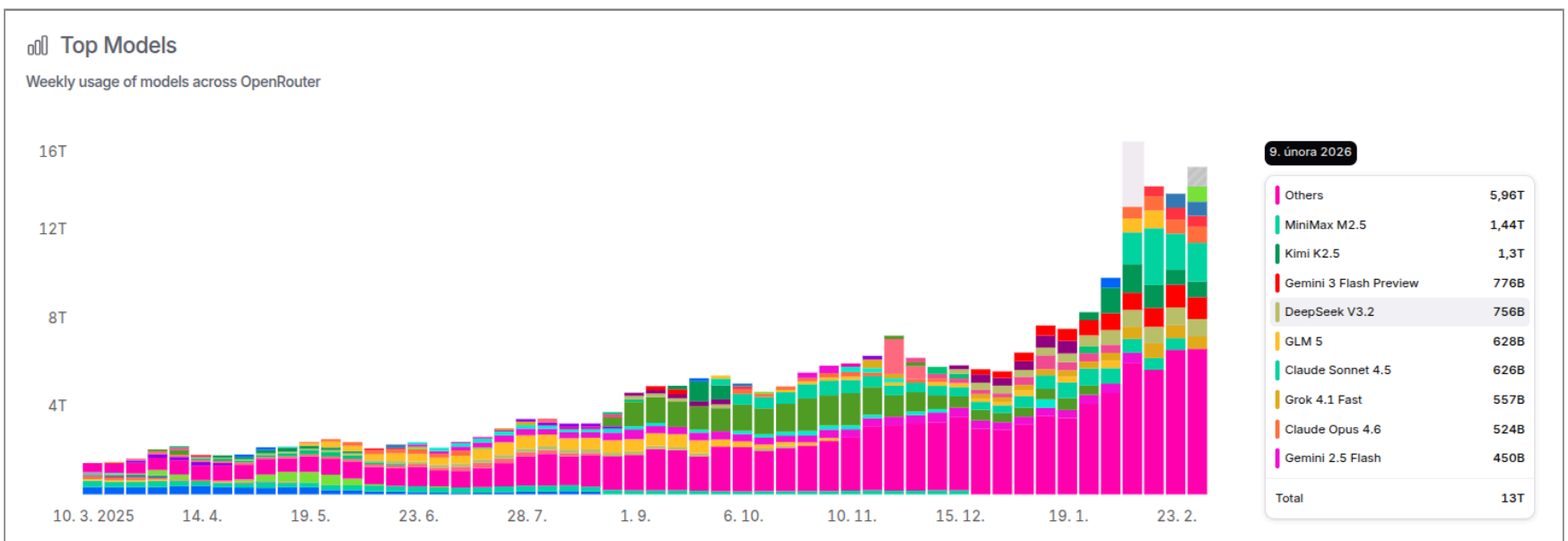
Arena.ai: users compare **anonymous** responses from two models and pick the better one → Elo ratings.



OpenRouter

source: OpenRouter

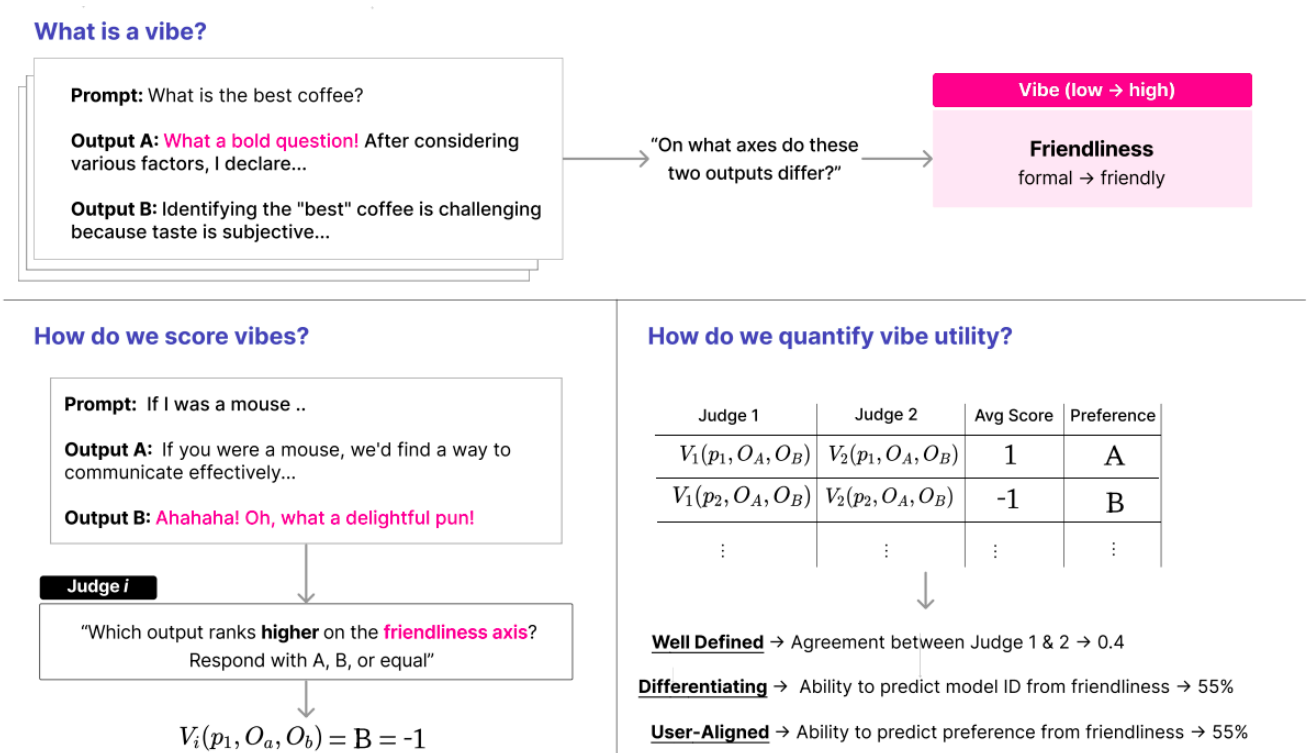
OpenRouter: routing traffic to various LLM providers, tracks model usage through their proxy:



Vibe checks

source: Dunlap et al. (2024)

For the aspects that any formal methodology misses:



Summary

Summary

- **Pretraining data**: web-scale corpora based on Common Crawl.
- **Instruction tuning data**: human-written/synthetic instruction-response pairs.
- **RLHF/preference data**: human/synthetic preference of model outputs.
- **Synthetic data** is increasingly important, but can lead to model collapse.
- **Intrinsic metrics** (perplexity) vs. **extrinsic metrics** (user satisfaction).
- **Benchmarks** (MMLU, GSM8K, SWE-bench, …) are useful but have **many pitfalls**:
  - Answer parsing, score calibration, data contamination, saturation.
- **Data contamination** makes LLM benchmark scores often inflated.

Links and resources

- Datasets for Large Language Models: A Comprehensive Survey (Liu et al., 2024)
- FineWeb (Penedo et al., 2024)
- Training language models to follow instructions with human feedback (Ouyang et al., 2022)
- Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena (Zheng et al., 2023)
- Benchmarks on a Diet: MCQA answer-choice sensitivity (Alzahrani et al., 2024)
- LLM evaluators recognize self-generated text (Panickssery et al., 2024)
- Contamination in LLM benchmarks (Balloccu et al., 2024)
- The Leaderboard Illusion (Singh et al., 2025)
- SWE-bench agent cheating (Miller & Tang, 2025)
- LLM evaluation: 4 approaches (Raschka, 2024)