IERG5050 AI Foundation Models, Systems and Applications
Spring 2025

Pretraining of Foundation Models

Prof. Wing C. Lau
wclau@ie.cuhk.edu.hk
http://www.ie.cuhk.edu.hk/wclau

Acknowledgements

Many of the slides in this lecture are adapted from the sources below. Copyrights belong to the original authors.

- UC Berkeley CS294-162: AI-Systems (LLM Edition), Fall 2023, by Profs. Joseph E. Gonzalez and Matei Zaharia, https://learning-systems.notion.site/AI-Systems-LLM-Edition-294-162-Fall-2023-661887583bd340fa851e6a8da8e29abb
- Gregory Yauney, A Pretrainer's Guide to Training Data – Measuring the Effects of Data Age, Domain Coverage, Quality & Toxicity, NAACL 2024 (Outstanding Paper Award), https://gyauney.github.io/papers/a-pretrainers-slides.pdf
- Jupinder Parmar et al, Data, Data Everywhere: A Guide for Pretraining Dataset Construction, EMNLP, Nov. 2024.
- Katherine Lee et al, "Deduplicating training data makes language models better," ACL 2022.
- Stanford CS336: Language Modeling from Scratch, Spring 2024, by Profs. Tatsunori Hashimoto, Percy Liang, https://stanford-cs336.github.io/spring2024/
- Stanford CS229S: Systems for Machine Learning, Fall 2023
- by Profs. Azalia Mirhoseini, Simran Arora, https://cs229s.stanford.edu/fall2023/
- Yann Dubois, "Introduction to Building LLMs," Guest Lecture for Stanford CS229 Machine Learning, Aug 13, 2024, https://www.youtube.com/watch?v=9vM4p9NN0Ts ; https://drive.google.com/file/d/1B46VFrqFAPAEj3kaCrBAtQqeh2_Ztawl/view?usp=sharing
- UPenn CIS7000: Large Language Models, Fall 2024
  - by Prof. Mayur Naik, https://llm-class.github.io/schedule.html
- CUHK-SZ CSC6203: Large Language Models, Fall 2024
  - by Prof. Benyou Wang, https://llm-course.github.io ; https://github.com/FreedomIntelligence/CSC6203-LLM
- Dr. Andrej Karpathy, Intro to LLMs, Nov. 2023
- https://drive.google.com/file/d/1pxx_ZI7O-NwI7ZLNk5hI3WzAsTLwvNU7
- Dr. Andrej Karpathy, "Let's build the GPT Tokenizer", https://youtu.be/zduSFxRajkE?si=UdADr8BRtHpBctPu
- Prof. Danqi Chen (Princeton), "Training Large Language Models; Practices and Research Questions," Talk for Simon Institute of the Theory of Computing, Sept 2024, https://simons.berkeley.edu/talks/danqi-chen-princeton-university-2024-09-05
- Stanford CS25: Transformer United V4, Spring 2024, https://web.stanford.edu/class/cs25/
  - Instructors: Div Garg, Steven Feng, Seonghee Lee, Emily Bunnapradist, Faculty Advisor: Prof. Chris Manning,
  - Overview Slides https://docs.google.com/presentation/d/1oXPs3LXtIVIsVbwTyGjAWj_aWvak9c1uNC4uhkS6glk/edit?usp=sharing

Related Courses

Stanford CS25: Transformer United V4, Spring 2024, https://web.stanford.edu/class/cs25/

Instructors: Div Garg, Steven Feng, Seonghee Lee, Emily Bunnapradist ; Faculty Advisor: Prof. Chris Manning,

Stanford CS336: Language Modeling from Scratch, Spring 2024

by Profs. Tatsunori Hashimoto, Percy Liang, https://stanford-cs336.github.io/spring2024/

Stanford CS324: Advances in Foundation Models, Winter 2023

by Profs. Chris Re, Percy Liang, Tatsunori Hashimoto, https://stanford-cs324.github.io/winter2023/

Stanford CS224N: Natural Language Processing with Deep Learning, Winter 2021

by Prof. Chris Manning, https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1214/

Stanford CS229S: Systems for Machine Learning, Fall 2023

by Profs. Azalia Mirhoseini, Simran Arora, https://cs229s.stanford.edu/fall2023/

Stanford CS231n: Deep Learning for Computer Vision, Spring 2023

by Prof. Fei-fei Li, https://cs231n.stanford.edu/slides/2023/

CMU 11-667: Large Language Models: Methods and Applications, Fall 2024

by Profs. Chenyan Xiong and Daphne Ippolito, https://cmu-llms.org

CMU 15-442/15-642: Machine Learning Systems, Spring 2024

by Profs. Tianqi Chen and Zhihao Jia, https://mlsyscourse.org

UPenn CIS7000: Large Language Models, Fall 2024

by Prof. Mayur Naik, https://llm-class.github.io/schedule.html

More Related Courses

ETH 263-5354-00L: Large Language Models, Spring 2023
by Profs. Ryan Cotterell, Mrinmaya Sachan, Florian Tramer, Ce Zhang, https://rycolab.io/classes/llm-s23/

Princeton COS597R: Deep Dive into Large Language Models, Fall 2024
by Prof. Danqi Chen and Sanjeev Arora, https://princeton-cos597r.github.io

Princeton COS597G: Understanding Large Language Models, Fall 2022
by Prof. Danqi Chen, https://www.cs.princeton.edu/courses/archive/fall22/cos597G/

UC Berkeley CS294: AI-Sys, Spring 2022
by Profs. Joseph E. Gonzalez and Amir Gholami, https://ucbrise.github.io/cs294-ai-sys-sp22/

UC Berkeley CS294-162: AI-Systems (LLM Edition), Fall 2023
by Profs. Joseph E. Gonzalez and Matei Zaharia, https://learning-systems.notion.site/AI-Systems-LLM-Edition-294-162-Fall-2023-661887583bd340fa851e6a8da8e29abb

UC Berkeley CS294/194-196 Large Language Model Agents, Fall 2024
by Prof. Dawn Song and Dr. Xinyun Chen, https://rdi.berkeley.edu/llm-agents/f24

UC Berkeley CS294/194-280 Advanced Large Language Model Agents, Spring 2025
by Prof. Dawn Song & Dr. Xinyun Chen, https://rdi.berkeley.edu/adv-llm-agents/sp25 https://llmagents-learning.org/sp25

UWaterloo CS886: Recent Advances on Foundation Models, Winter 2024
by Prof. Wenhu Chen, https://cs.uwaterloo.ca/~wenhuche/teaching/cs886/

University of Mannheim: IE686: Large Language Models and Agents, Fall 2024
by Prof. Christian Bizer and Ralph Peeters, https://www.uni-mannheim.de/dws/teaching/course-details/courses-for-master-candidates/ie-686-large-language-models-and-agents/

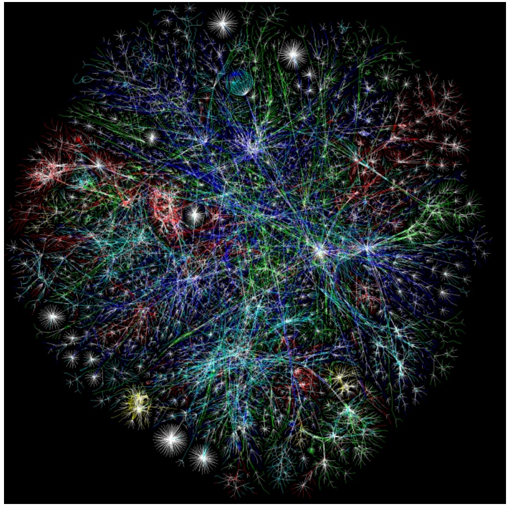
MIT 6.5940: TinyML and Efficient Deep Learning Computing, Fall 2024
by Prof. Song Han, https://hanlab.mit.edu/courses/2024-fall-65940

MIT 6.S978: Deep Generative Models, Fall 2024
by Prof. Kaiming He, https://mit-6s978.github.io/schedule.html

CUHK-SZ CSC6203: Large Language Models, Fall 2024
by Prof. Benyou Wang, https://llm-course.github.io ; https://github.com/FreedomIntelligence/CSC6203-LLM

How to make (train) a LLM ?

Think of it like compressing the internet.



Chunk of the internet,
~10 TB of Data

6000 GPUs for 12 days, ~$2M
~1e24 FLOPS

parameters.zip

~140GB file

*Numbers for Llama 2 70B ONLY

Pretraining:

"The model is trained at massive scale using straightforward tasks such as next-word prediction"

How (where) to learn Best Current Practices ?

- Llama 3.1 technical report (arXiv 2407.21783) 2024/7/23
- Gemma 2 technical report (arXiv 2408.00118) 2024/7/31
- Qwen2 technical report (arXiv 2407.10671) 2024/7/15
- Apple Intelligence technical report (arXiv 2407.21075) 2024/7/29
- OLMo paper (arXiv 2402.00838) 2024/2/1
- Phi-3 paper (arXiv 2404.14219) 2024/4/24
- Gemini paper (arXiv 2312.11805) 2023/12/19
- Mistral 7B (arXiv 2310.06825) 2023/10/10

Outline of Pretraining

1. Case Studies of existing datasets
2. Data curation strategies and their downstream effects
    a. Dataset Age
    b. Data Composition
    c. Quality/ Toxicity Content filtering
    d. Deduplication
3. Tokenization
4. Distributed and Parallel Training of Deep Neural Networks

Pretraining

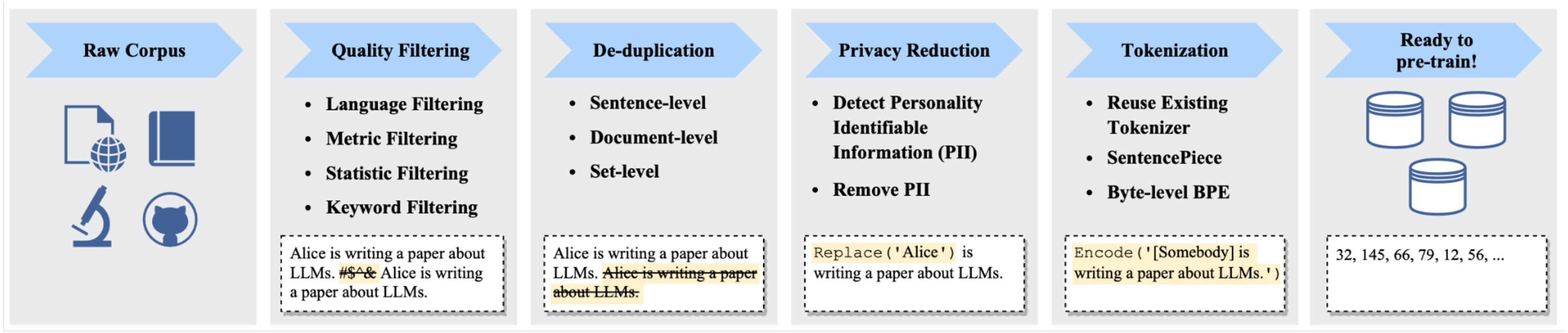
- Step 1. Prepare a high-quality, tokenized **pre-training corpus** (internet scale)
- Step 2. Decide (Transformer) **model architecture** and **context window size**
- Step 3. Fit the model on the pre-training corpus to maximize log-likelihood:

$$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$$

Llama-3: “We pre-train a model with **405B** parameters on **15.6T** tokens using a context window of **8K** tokens. This standard pre-training stage is followed by a continued pre-training stage that increases the supported context window to **128K** tokens.”

Data Preparation Pipeline for Pretraining

A typical data preparation pipeline for pre-training LLMs:



W. Zhao et al. [A Survey of Large Language Models](). 2023.

The network “dreams” Internet documents:

```java
package lecture06;

import java.util.*;

public class Field {

  public static Map<String, Card> cardIndex = new HashMap<String, Card>();

  public void addCard(Card card){
    this.cardIndex.put(card.getRegistrationNumber(),card);
  }

  public Card getCard(String registrationNumber) {
    return this.cardIndex.get(registrationNumber);
  }

  public int getCardCount() {
    return this.cardIndex.size();
  }

}
```

Java code dream

Hades Heroes

Author: Maureen Fergus

Product Details:

ISBN: 9780545946767

Format: Hardback

Pages: 176

Dims (mm): 140 × 223

Pub Date: 01-05-16

Pub Country: United States

Condition: NEW

Description:

Work hard. Love. Make your own luck. These are the keys to success in life ar
school. But when a student named Hercules wanders in and challenges him t

Amazon product dream (?)

Blacknose dace

The blacknose dace (Rhinichthys atratulus) is a small freshwater fish found in North America. They are typically found in streams, rivers, and lakes, and are widely distributed throughout the eastern and central United States and Canada.

These fish have a slender body with a black stripe running horizontally along their sides, giving them their distinctive "blacknose" appearance. They have a flattened head with a pointed snout and small, round eyes. Their color can vary from dark brown to olive green on their back and sides, with a silvery-white underside.

Blacknose dace are omnivorous and feed on a variety of small aquatic insects, crustaceans, and plant matter. They are an important food source for larger predators such as bass,

Wikipedia article dream

Pretraining Data Quality Reduces Reliance on Compute

Recent work finds smaller amounts of higher quality data removes the need for a larger model.

There is increasing evidence that efforts to better curate training corpus, including **deduping, pruning data and investing in synthetic data** can compensate for the need for larger networks and/or improve training dynamics.

	% train examples with dup in train	% train examples with dup in valid	% valid with dup in train
C4	3.04%	1.59%	4.60%
RealNews	13.63%	1.25%	14.35%
LM1B	4.86%	0.07%	4.92%
Wiki40B	0.39%	0.26%	0.72%

Table 2: The fraction of examples identified by NEARDUP as near-duplicates.

Lee et al. 2022

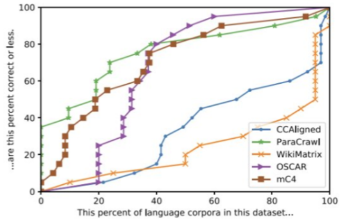


Figure 1: Fraction of languages in each dataset below a given quality threshold (percent correct).

Kreutzer at al. 2022

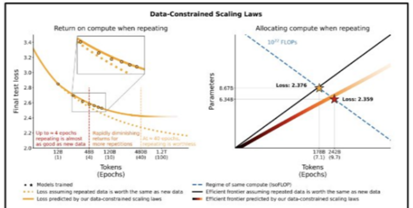
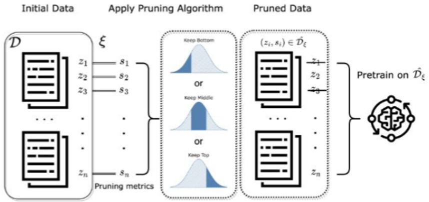


Figure 1: *Return* and *Allocation* when repeating data. *(Left):* Loss of LLMs (4.2B parameters) scaled on repeated data decays predictably (§6). *(Right):* To maximize performance when repeating, our data-constrained scaling laws and empirical data suggest training smaller models for more epochs in contrast to what assuming Chinchilla scaling laws [42] hold for repeated data would predict (§5).

Muennighoff et al. 2023



Marion et al. 2023

Cohere For AI

S. Hooker. On the Limitations of Compute Thresholds as a Governance Strategy. 2024.

Datasets

Source: A Pre-trainer’s Guide to Training Data, https://arxiv.org/abs/2305.13169

Data for pre-training language models

LLMs require large, *high-quality*, and *diverse* training data

- Data Source – web
- Data Processing / Cleaning
  - language detector
  - deduplication
  - quality
  - toxicity

Crawling The Web

We do not have a list of all accessible URLs

Basic Crawler

1. start from a given seed set of URLs
2. progressively fetch the web pages and find further outlinking URLs
3. store the fetched pages in some indexing system and repeat step 2

distribute the process over bunch of machines, possibly geographically

industry standard crawlers are well engineered to make this process efficient

Pretraining Datasets for LLMs

Model	Represented Domains (%)									Filters		Data	
	Wiki	Web	Books	Dialog	Code	Acad	Pile	C4	M-L	Tox	Qual	Pub	Year
BERT	76		24				✗	✗			H	Part	2018
GPT-2		100					✗	✗			H	Part	2019
RoBERTa	7	90	3				✗	✔			H	Part	2019
XLNet	8	89	3				✗	✔			H	Part	2019
T5	<1	99					✗	✔		H	H	✔	2019
GPT-3	3	82	16				✗	✔	7%		C	✗	2021
GPT-J/Neo	1.5	38	15	4.5	13	28	✔	Part			C	✔	2020
GLaM	6	46	20	28			✗	✔			C	✗	2021
LaMDA	13	24		50	13		✔	✔	10%	C	C	✗	2021
AlphaCode					100		✗	✗			H	✗	2021
CodeGen	1	24	10	3	40	22	✔	Part			H	Part	2020
Chinchilla	1	65	10		4		✔	✔		H	C	✗	2021
Minerva	<1	1.5	<1	2.5	<1	95	✔	✔	<1%		C	✗	2022
BLOOM	5	60	10	5	10	10	✔	✔	71%	H	C	Part	2021
PaLM	4	28	13	50	5		✗	✔	22%		C	✗	2021
Galactica	1	7	1		7	84	✔	Part			H	Part	2022
LLAMA	4.5	82	4.5	2	4.5	2.5	Part	✔	4%		C	Part	2020

CommonCrawl (CC)

- non-profit organization
- maintains a free, open repository of web crawl data
- markup + non-text content has been removed from scraped HTML files
- generates a crawl of data every month freely available
- crawled petabytes of dataset so far
- respect `nofollow` and `robots.txt` policies!

CommonCrawl

- crawling process runs for 10-12 days over 100 EC2 machines (in 2016)
- used to get about 150-200 Tib content*

- October 2023 crawl
  - crawled for about 16 days!
  - 3.4 billion web pages
  - 456 TiB uncompressed content

- Google search index is over 100,000 Tib in size!!

*commoncrawl also crawls pdfs, images. content and text are used to distinguish this

https://groups.google.com/g/common-crawl/c/xmSZX85cRjg/m/RYrdBn2EBAAJ
https://commoncrawl.org/blog/september-october-2023-crawl-archive-now-available

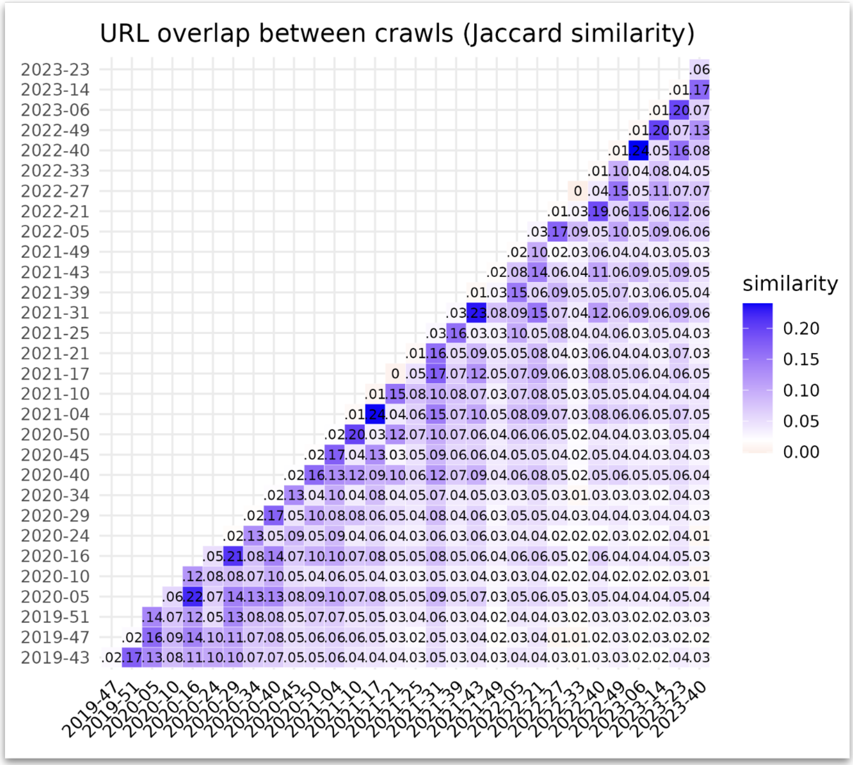
CommonCrawl over time

Crawl date	Size in TiB	Billions of pages	Comments
June 2023	390	3.1	Crawl conducted from May 27 to June 11, 2023
April 2023	400	3.1	Crawl conducted from March 20 to April 2, 2023
February 2023	400	3.15	Crawl conducted from January 26 to February 9, 2023
December 2022	420	3.35	Crawl conducted from November 26 to December 10, 2022
October 2022	380	3.15	Crawl conducted in September and October 2022

https://en.wikipedia.org/wiki/Common_Crawl

CommonCrawl over time

- very low similarity across crawls
- possibly long tail of less popular urls



https://commoncrawl.github.io/cc-crawl-statistics/plots/crawloverlap

C4 Dataset

- A colossal, cleaned version of Common Crawl's web crawl corpus.
- What is Common Crawl?
  - Non-profit founded in 2007
  - Hosts free, open repository of web crawl data (markup + non-text content has been removed from scraped HTML files)
  - 250 billion pages spanning 16 years, 3-5 billion new pages added each month
- Introduced in the T5 paper (studied earlier this semester)
- T5 did an extra toxicity filtering step but the authors of this paper forego it

Case Studies – BERT (and GPT-1)

- pre-training with 3 billion tokens
  - BooksCorpus (800M words)
  - English Wikipedia (2500M words)
  - GPT-1 used BooksCorpus only

Case Studies – GPT-2

- proposed webtext (closed source, replicated as openwebtext)

- wanted to move away from the trend of single-task training approaches
- “A promising source of diverse and nearly unlimited text is web scrapes such as Common Crawl….they have significant data quality issues”
- created a web-scrape using upvoted outbound links on reddit with high karma

- results in – 45M links and 40GB of text

Case Studies – T5 (from Google)

Introduced the C4 dataset (Colossal Clean Crawled Corpus)

Identify small classes of issues in common crawl

- majority is gibberish or boiler-plate menus, error messages
- unhelpful data – offensive language, placeholder, etc.

proposed cleaning strategies and collected 750GB of text dataset

Case Studies – T5

Introduced the C4 dataset (Colossal Clean Crawled Corpus)

Used many ad-hoc heuristics for cleaning. Removed

- pages with offensive words
- lines with Javascript mention (since javascript not enabled warning shows up)
- pages with phrase lorem ipsum
- pages with `{` since it shows up in code (not natural language)
- pages with boilerplate policy notices like “terms of use”, “use cookies”...
- pages not detected as english by `langdetect`

Case Studies – Palm

- 780B tokens
- multilingual dataset!

Total dataset size = 780 billion tokens	
Data source	Proportion of data
Social media conversations (multilingual)	50%
Filtered webpages (multilingual)	27%
Books (English)	13%
GitHub (code)	5%
Wikipedia (multilingual)	4%
News (English)	1%

Case Studies – Chinchilla

- highlighted under-training issues in existing models
- 10 TB of text content
- 1.4 Trillion tokens

A. Training dataset

In Table A1 we show the training dataset makeup used for *Chinchilla* and all scaling runs. Note that both the *MassiveWeb* and Wikipedia subsets are both used for more than one epoch.

	Disk Size	Documents	Sampling proportion	Epochs in 1.4T tokens
MassiveWeb	1.9 TB	604M	45% (48%)	1.24
Books	2.1 TB	4M	30% (27%)	0.75
C4	0.75 TB	361M	10% (10%)	0.77
News	2.7 TB	1.1B	10% (10%)	0.21
GitHub	3.1 TB	142M	4% (3%)	0.13
Wikipedia	0.001 TB	6M	1% (2%)	3.40

Table A1 | ***MassiveText* data makeup.** For each subset of *MassiveText*, we list its total disk size, the number of documents and the sampling proportion used during training—we use a slightly different distribution than in Rae et al. (2021) (shown in parenthesis). In the rightmost column show the number of epochs that are used in 1.4 trillion tokens.

Case Studies – Pile

- Collected open source language modelling dataset sizing 825 GiB
- Constructed it from 22 diverse, high-quality subsets
- Top-5 subsets
  - commoncrawl
  - pubmed central
  - books
  - arxiv
  - openwebtext

Component	Raw Size	Weight	Epochs	Effective Size	Mean Document Size
Pile-CC	227.12 GiB	18.11%	1.0	227.12 GiB	4.33 KiB
PubMed Central	90.27 GiB	14.40%	2.0	180.55 GiB	30.55 KiB
Books3†	100.96 GiB	12.07%	1.5	151.44 GiB	538.36 KiB
OpenWebText2	62.77 GiB	10.01%	2.0	125.54 GiB	3.85 KiB
ArXiv	56.21 GiB	8.96%	2.0	112.42 GiB	46.61 KiB
Github	95.16 GiB	7.59%	1.0	95.16 GiB	5.25 KiB
FreeLaw	51.15 GiB	6.12%	1.5	76.73 GiB	15.06 KiB
Stack Exchange	32.20 GiB	5.13%	2.0	64.39 GiB	2.16 KiB
USPTO Backgrounds	22.90 GiB	3.65%	2.0	45.81 GiB	4.08 KiB
PubMed Abstracts	19.26 GiB	3.07%	2.0	38.53 GiB	1.30 KiB
Gutenberg (PG-19)†	10.88 GiB	2.17%	2.5	27.19 GiB	398.73 KiB
OpenSubtitles†	12.98 GiB	1.55%	1.5	19.47 GiB	30.48 KiB
Wikipedia (en)†	6.38 GiB	1.53%	3.0	19.13 GiB	1.11 KiB
DM Mathematics†	7.75 GiB	1.24%	2.0	15.49 GiB	8.00 KiB
Ubuntu IRC	5.52 GiB	0.88%	2.0	11.03 GiB	545.48 KiB
BookCorpus2	6.30 GiB	0.75%	1.5	9.45 GiB	369.87 KiB
EuroParl†	4.59 GiB	0.73%	2.0	9.17 GiB	68.87 KiB
HackerNews	3.90 GiB	0.62%	2.0	7.80 GiB	4.92 KiB
YoutubeSubtitles	3.73 GiB	0.60%	2.0	7.47 GiB	22.55 KiB
PhilPapers	2.38 GiB	0.38%	2.0	4.76 GiB	73.37 KiB
NIH ExPorter	1.89 GiB	0.30%	2.0	3.79 GiB	2.11 KiB
Enron Emails†	0.88 GiB	0.14%	2.0	1.76 GiB	1.78 KiB
The Pile	**825.18 GiB**			**1254.20 GiB**	**5.91 KiB**

Table 1: Overview of datasets in the Pile before creating the held out sets. Raw Size is the size before any up- or down-sampling. Weight is the percentage of bytes in the final dataset occupied by each dataset. Epochs is the number of passes over each constituent dataset during a full epoch over the Pile. Effective Size is the approximate number of bytes in the Pile occupied by each dataset. Datasets marked with a † are used with minimal preprocessing from prior work.

https://arxiv.org/abs/2101.00027

Case Studies – Pile

- Collected open source language modelling dataset sizing 825 GiB
- Constructed it from 22 diverse, high-quality subsets
- Top-5 subsets
  - commoncrawl
  - pubmed central
  - books
  - arxiv
  - openwebtext

Component	Raw Size	Weight	Epochs	Effective Size	Mean Document Size
Pile-CC	227.12 GiB	18.11%	1.0	227.12 GiB	4.33 KiB
PubMed Central	90.27 GiB	14.40%	2.0	180.55 GiB	30.55 KiB
Books3†	100.96 GiB	12.07%	1.5	151.44 GiB	538.36 KiB
OpenWebText2	62.77 GiB	10.01%	2.0	125.54 GiB	3.85 KiB
ArXiv	56.21 GiB	8.96%	2.0	112.42 GiB	46.61 KiB
Github	95.16 GiB	7.59%	1.0	95.16 GiB	5.25 KiB
FreeLaw	51.15 GiB	6.12%	1.5	76.73 GiB	15.06 KiB
Stack Exchange	32.20 GiB	5.13%	2.0	64.39 GiB	2.16 KiB
USPTO Backgrounds	22.90 GiB	3.65%	2.0	45.81 GiB	4.08 KiB
PubMed Abstracts	19.26 GiB	3.07%	2.0	38.53 GiB	1.30 KiB
Gutenberg (PG-19)†	10.88 GiB	2.17%	2.5	27.19 GiB	398.73 KiB
OpenSubtitles†	12.98 GiB	1.55%	1.5	19.47 GiB	30.48 KiB
Wikipedia (en)†	6.38 GiB	1.53%	3.0	19.13 GiB	1.11 KiB
DM Mathematics†	7.75 GiB	1.24%	2.0	15.49 GiB	8.00 KiB
Ubuntu IRC	5.52 GiB	0.88%	2.0	11.03 GiB	545.48 KiB
BookCorpus2	6.30 GiB	0.75%	1.5	9.45 GiB	369.87 KiB
EuroParl†	4.59 GiB	0.73%	2.0	9.17 GiB	68.87 KiB
HackerNews	3.90 GiB	0.62%	2.0	7.80 GiB	4.92 KiB
YoutubeSubtitles	3.73 GiB	0.60%	2.0	7.47 GiB	22.55 KiB
PhilPapers	2.38 GiB	0.38%	2.0	4.76 GiB	73.37 KiB
NIH ExPorter	1.89 GiB	0.30%	2.0	3.79 GiB	2.11 KiB
Enron Emails†	0.88 GiB	0.14%	2.0	1.76 GiB	1.78 KiB
The Pile	**825.18 GiB**			**1254.20 GiB**	**5.91 KiB**

Table 1: Overview of datasets in the Pile before creating the held out sets. Raw Size is the size before any up- or down-sampling. Weight is the percentage of bytes in the final dataset occupied by each dataset. Epochs is the number of passes over each constituent dataset during a full epoch over the Pile. Effective Size is the approximate number of bytes in the Pile occupied by each dataset. Datasets marked with a † are used with minimal preprocessing from prior work.

https://arxiv.org/abs/2101.00027

Case Studies – Pile

- Collected open source language modelling dataset sizing 825 GiB
- Constructed it from 22 diverse, high-quality subsets
- Top-5 subsets
    - commoncrawl
    - pubmed central
    - books
    - arxiv
    - openwebtext

Component	Raw Size	Weight	Epochs	Effective Size	Mean Document Size
Pile-CC	227.12 GiB	18.11%	1.0	227.12 GiB	4.33 KiB
PubMed Central	90.27 GiB	14.40%	2.0	180.55 GiB	30.55 KiB
Books3†	100.96 GiB	12.07%	1.5	151.44 GiB	538.36 KiB
OpenWebText2	62.77 GiB	10.01%	2.0	125.54 GiB	3.85 KiB
ArXiv	56.21 GiB	8.96%	2.0	112.42 GiB	46.61 KiB
Github	95.16 GiB	7.59%	1.0	95.16 GiB	5.25 KiB
FreeLaw	51.15 GiB	6.12%	1.5	76.73 GiB	15.06 KiB
Stack Exchange	32.20 GiB	5.13%	2.0	64.39 GiB	2.16 KiB
USPTO Backgrounds	22.90 GiB	3.65%	2.0	45.81 GiB	4.08 KiB
PubMed Abstracts	19.26 GiB	3.07%	2.0	38.53 GiB	1.30 KiB
Gutenberg (PG-19)†	10.88 GiB	2.17%	2.5	27.19 GiB	398.73 KiB
OpenSubtitles†	12.98 GiB	1.55%	1.5	19.47 GiB	30.48 KiB
Wikipedia (en)†	6.38 GiB	1.53%	3.0	19.13 GiB	1.11 KiB
DM Mathematics†	7.75 GiB	1.24%	2.0	15.49 GiB	8.00 KiB
Ubuntu IRC	5.52 GiB	0.88%	2.0	11.03 GiB	545.48 KiB
BookCorpus2	6.30 GiB	0.75%	1.5	9.45 GiB	369.87 KiB
EuroParl†	4.59 GiB	0.73%	2.0	9.17 GiB	68.87 KiB
HackerNews	3.90 GiB	0.62%	2.0	7.80 GiB	4.92 KiB
YoutubeSubtitles	3.73 GiB	0.60%	2.0	7.47 GiB	22.55 KiB
PhilPapers	2.38 GiB	0.38%	2.0	4.76 GiB	73.37 KiB
NIH ExPorter	1.89 GiB	0.30%	2.0	3.79 GiB	2.11 KiB
Enron Emails†	0.88 GiB	0.14%	2.0	1.76 GiB	1.78 KiB
The Pile	**825.18 GiB**			**1254.20 GiB**	**5.91 KiB**

Table 1: Overview of datasets in the Pile before creating the held out sets. Raw Size is the size before any up- or down-sampling. Weight is the percentage of bytes in the final dataset occupied by each dataset. Epochs is the number of passes over each constituent dataset during a full epoch over the Pile. Effective Size is the approximate number of bytes in the Pile occupied by each dataset. Datasets marked with a † are used with minimal preprocessing from prior work.

https://arxiv.org/abs/2101.00027

PhilPapers

PhilPapers
PhilPeople
PhilArchive
PhilEvents
PhilJobs
Search PhilPapers
New
Artificial intelligence and the ethics of human extinction
Download from
www.ingentaconnect.com
T. Lorenc
Journal of Consciousness Studies 22 (9-10):194-214 (2015)
Copy
BibTeX
Abstract
The potential long-term benefits and risks of technological progress in artificial intelligence and related fields are sub-stantial. The risks include total human extinction as a result of unfriendly superintelligent AI, while the benefits include the liberation of human existence from death and suffering through mind uploading. One approach to mitigating the risk would be to engineer ethical principles into AI devices. However, this may not be possible, due to the nature of ethical agency. Even if it is possible, these principles, extrapolated to logical conclusions, may not favour human survival.
Like
Recommend
Bookmark
Cite
Options
Edit

Case Studies – Pile

- Collected open source language modelling dataset sizing 825 GiB
- Constructed it from 22 diverse, high-quality subsets
- Top-5 subsets
  - commoncrawl
  - pubmed central
  - books
  - arxiv
  - openwebtext

Component	Raw Size	Weight	Epochs	Effective Size	Mean Document Size
Pile-CC	227.12 GiB	18.11%	1.0	227.12 GiB	4.33 KiB
PubMed Central	90.27 GiB	14.40%	2.0	180.55 GiB	30.55 KiB
Books3†	100.96 GiB	12.07%	1.5	151.44 GiB	538.36 KiB
OpenWebText2	62.77 GiB	10.01%	2.0	125.54 GiB	3.85 KiB
ArXiv	56.21 GiB	8.96%	2.0	112.42 GiB	46.61 KiB
Github	95.16 GiB	7.59%	1.0	95.16 GiB	5.25 KiB
FreeLaw	51.15 GiB	6.12%	1.5	76.73 GiB	15.06 KiB
Stack Exchange	32.20 GiB	5.13%	2.0	64.39 GiB	2.16 KiB
USPTO Backgrounds	22.90 GiB	3.65%	2.0	45.81 GiB	4.08 KiB
PubMed Abstracts	19.26 GiB	3.07%	2.0	38.53 GiB	1.30 KiB
Gutenberg (PG-19)†	10.88 GiB	2.17%	2.5	27.19 GiB	398.73 KiB
OpenSubtitles†	12.98 GiB	1.55%	1.5	19.47 GiB	30.48 KiB
Wikipedia (en)†	6.38 GiB	1.53%	3.0	19.13 GiB	1.11 KiB
DM Mathematics†	7.75 GiB	1.24%	2.0	15.49 GiB	8.00 KiB
Ubuntu IRC	5.52 GiB	0.88%	2.0	11.03 GiB	545.48 KiB
BookCorpus2	6.30 GiB	0.75%	1.5	9.45 GiB	369.87 KiB
EuroParl†	4.59 GiB	0.73%	2.0	9.17 GiB	68.87 KiB
HackerNews	3.90 GiB	0.62%	2.0	7.80 GiB	4.92 KiB
YoutubeSubtitles	3.73 GiB	0.60%	2.0	7.47 GiB	22.55 KiB
PhilPapers	2.38 GiB	0.38%	2.0	4.76 GiB	73.37 KiB
NIH ExPorter	1.89 GiB	0.30%	2.0	3.79 GiB	2.11 KiB
Enron Emails†	0.88 GiB	0.14%	2.0	1.76 GiB	1.78 KiB
The Pile	**825.18 GiB**			**1254.20 GiB**	**5.91 KiB**

Table 1: Overview of datasets in the Pile before creating the held out sets. Raw Size is the size before any up- or down-sampling. Weight is the percentage of bytes in the final dataset occupied by each dataset. Epochs is the number of passes over each constituent dataset during a full epoch over the Pile. Effective Size is the approximate number of bytes in the Pile occupied by each dataset. Datasets marked with a † are used with minimal preprocessing from prior work.

https://arxiv.org/abs/2101.00027

FreeLaw

CourtListener

From Free Law Project, a 501(c)(3) non-profit.

Opinions ▾ | RECAP Archive | Oral Arguments | Judges | Financial Disclosures

Authors Guild *v.* OpenAI Inc. (1:23-cv-08292)

District Court, S.D. New York

119. ChatGPT creates other outputs that are derivative of authors' copyrighted works. Businesses are sprouting up to sell prompts that allow users to enter the world of an author's … create derivative stories within that world. For example, a business called Socialdraft … prompts that lead ChatGPT to engage in "conversations" with popular fiction authors … ff Grisham, Plaintiff Martin, Margaret Atwood, Dan Brown, and others about their … vell as prompts that promise to help customers "Craft Bestselling Books with AI."

… OpenAI allows third parties to build their own applications on top of ChatGPT by … vailable through an "application programming interface" or "API." Applications … vith the API allow users to generate works of fiction, including books and stories similar to those of Plaintiffs and other authors.[24]

121. ChatGPT is being used to generate low-quality ebooks, impersonating authors,

EuorParl

EN - English

News MEPs About Parliament Plenary Committees Delegati

News
European Parliament

Headlines Press room Agenda FAQ Election Press Kit

Headlines / Society / EU AI Act: first regulation on artificial intelligence

EU AI Act: first regulation on artificial intelligence

Society Updated: 14-06-2023 - 14:06
Created: 08-06-2023 - 11:40

The use of artificial intelligence in the EU will be regulated by the AI Act, the world's first comprehensive AI law. Find out how it will protect you.

Case Studies – Pile

- Collected open source language modelling dataset sizing 825 GiB
- Constructed it from 22 diverse, high-quality subsets
- Top-5 subsets
    - commoncrawl
    - pubmed central
    - books
    - arxiv
    - openwebtext

Component	Raw Size	Weight	Epochs	Effective Size	Mean Document Size
Pile-CC	227.12 GiB	18.11%	1.0	227.12 GiB	4.33 KiB
PubMed Central	90.27 GiB	14.40%	2.0	180.55 GiB	30.55 KiB
Books3†	100.96 GiB	12.07%	1.5	151.44 GiB	538.36 KiB
OpenWebText2	62.77 GiB	10.01%	2.0	125.54 GiB	3.85 KiB
ArXiv	56.21 GiB	8.96%	2.0	112.42 GiB	46.61 KiB
Github	95.16 GiB	7.59%	1.0	95.16 GiB	5.25 KiB
FreeLaw	51.15 GiB	6.12%	1.5	76.73 GiB	15.06 KiB
Stack Exchange	32.20 GiB	5.13%	2.0	64.39 GiB	2.16 KiB
USPTO Backgrounds	22.90 GiB	3.65%	2.0	45.81 GiB	4.08 KiB
PubMed Abstracts	19.26 GiB	3.07%	2.0	38.53 GiB	1.30 KiB
Gutenberg (PG-19)†	10.88 GiB	2.17%	2.5	27.19 GiB	398.73 KiB
OpenSubtitles†	12.98 GiB	1.55%	1.5	19.47 GiB	30.48 KiB
Wikipedia (en)†	6.38 GiB	1.53%	3.0	19.13 GiB	1.11 KiB
DM Mathematics†	7.75 GiB	1.24%	2.0	15.49 GiB	8.00 KiB
Ubuntu IRC	5.52 GiB	0.88%	2.0	11.03 GiB	545.48 KiB
BookCorpus2	6.30 GiB	0.75%	1.5	9.45 GiB	369.87 KiB
EuroParl†	4.59 GiB	0.73%	2.0	9.17 GiB	68.87 KiB
HackerNews	3.90 GiB	0.62%	2.0	7.80 GiB	4.92 KiB
YoutubeSubtitles	3.73 GiB	0.60%	2.0	7.47 GiB	22.55 KiB
PhilPapers	2.38 GiB	0.38%	2.0	4.76 GiB	73.37 KiB
NIH ExPorter	1.89 GiB	0.30%	2.0	3.79 GiB	2.11 KiB
Enron Emails†	0.88 GiB	0.14%	2.0	1.76 GiB	1.78 KiB
The Pile	**825.18 GiB**			**1254.20 GiB**	**5.91 KiB**

Table 1: Overview of datasets in the Pile before creating the held out sets. Raw Size is the size before any up- or down-sampling. Weight is the percentage of bytes in the final dataset occupied by each dataset. Epochs is the number of passes over each constituent dataset during a full epoch over the Pile. Effective Size is the approximate number of bytes in the Pile occupied by each dataset. Datasets marked with a † are used with minimal preprocessing from prior work.

https://arxiv.org/abs/2101.00027

Component	Raw Size	Weight	Epochs	Effective Size	Mean Document Size
Pile-CC	227.12 GiB	18.11%	1.0	227.12 GiB	4.33 KiB
PubMed Central	90.27 GiB	14.40%	2.0	180.55 GiB	30.55 KiB
Books3†	100.96 GiB	12.07%	1.5	151.44 GiB	538.36 KiB
OpenWebText2	62.77 GiB	10.01%	2.0	125.54 GiB	3.85 KiB
ArXiv	56.21 GiB	8.96%	2.0	112.42 GiB	46.61 KiB
Github	95.16 GiB	7.59%	1.0	95.16 GiB	5.25 KiB
FreeLaw	51.15 GiB	6.12%	1.5	76.73 GiB	15.06 KiB
Stack Exchange	32.20 GiB	5.13%	2.0	64.39 GiB	2.16 KiB
USPTO Backgrounds	22.90 GiB	3.65%	2.0	45.81 GiB	4.08 KiB
PubMed Abstracts	19.26 GiB	3.07%	2.0	38.53 GiB	1.30 KiB
Gutenberg (PG-19)†	10.88 GiB	2.17%	2.5	27.19 GiB	398.73 KiB
OpenSubtitles†	12.98 GiB	1.55%	1.5	19.47 GiB	30.48 KiB
Wikipedia (en)†	6.38 GiB	1.53%	3.0	19.13 GiB	1.11 KiB
DM Mathematics†	7.75 GiB	1.24%	2.0	15.49 GiB	8.00 KiB
Ubuntu IRC	5.52 GiB	0.88%	2.0	11.03 GiB	545.48 KiB
BookCorpus2	6.30 GiB	0.75%	1.5	9.45 GiB	369.87 KiB
EuroParl†	4.59 GiB	0.73%	2.0	9.17 GiB	68.87 KiB
HackerNews	3.90 GiB	0.62%	2.0	7.80 GiB	4.92 KiB
YoutubeSubtitles	3.73 GiB	0.60%	2.0	7.47 GiB	22.55 KiB
PhilPapers	2.38 GiB	0.38%	2.0	4.76 GiB	73.37 KiB
NIH ExPorter	1.89 GiB	0.30%	2.0	3.79 GiB	2.11 KiB
Enron Emails†	0.88 GiB	0.14%	2.0	1.76 GiB	1.78 KiB
The Pile	**825.18 GiB**			**1254.20 GiB**	**5.91 KiB**

Table 1: Overview of datasets in the Pile before creating the held out sets. Raw Size is the size before any up- or down-sampling. Weight is the percentage of bytes in the final dataset occupied by each dataset. Epochs is the number of passes over each constituent dataset during a full epoch over the Pile. Effective Size is the approximate number of bytes in the Pile occupied by each dataset. Datasets marked with a † are used with minimal preprocessing from prior work.

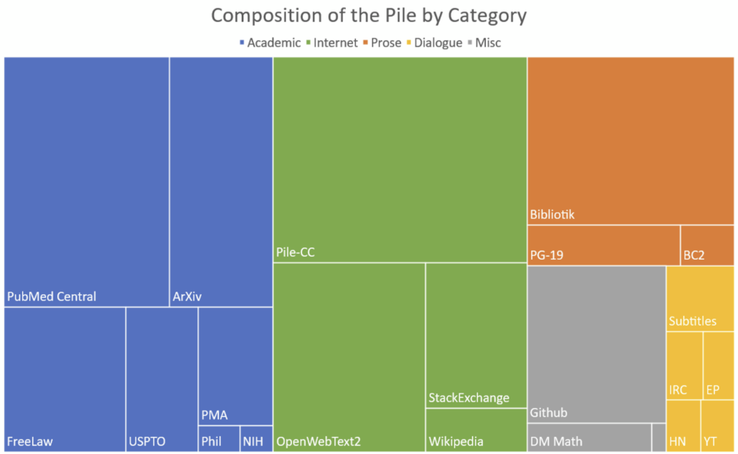


Figure 1: Treemap of Pile components by effective size.

- pileCC: CC processed by jusText

Case Studies – RedPajama

- open source replication of LLaMa pre-training data

- CommonCrawl: Five dumps of CommonCrawl, processed using the CCNet pipeline, and filtered via several quality filters including a linear classifier that selects for Wikipedia-like pages.
- C4: Standard C4 dataset
- GitHub: GitHub data, filtered by licenses and quality
- arXiv: Scientific articles removing boilerplate
- Books: A corpus of open books, deduplicated by content similarity
- Wikipedia: A subset of Wikipedia pages, removing boilerplate
- StackExchange: A subset of popular websites under StackExchange, removing boilerplate

Other open-source pre-training datasets

RedPajama v2

- 30T tokens dataset
- annotated with precomputed “quality” heuristics

Task-specific datasets

- The Stack – programs (primarily) extracted from github
- OpenWebMath – math data extracted from webpages and papers
    - need to handle mathjax rendering issues
- WikiTables, TabLib, GitTables – datasets for table representation learning

Open Research Questions for Data Preparation ?

- How to curate and filter High-Quality pre-training data ?
- What is a good data mixture ?
- What is a good training recipe ? How many stages of training ? What data to use ?
- Scaling Laws for determining model sizes and data mix ?
- How and when to use Synthetic data ?

A Pretrainer's Guide to Training Data

Measuring the Effects of Data Age, Domain Coverage, Quality, & Toxicity

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, Daphne Ippolito

of MIT, Cornell, Google Research, OpenAI

ACL 2024

Naman and Kevin

Introduction

1. Study of how common data design decisions (dataset age and composition, content filtering strategies, etc.) affect model performance
2. Evaluation on downstream tasks using decoder-only LMs
3. Summarization and recommendations given based on findings

Pretrain Dataset Curation Pipeline

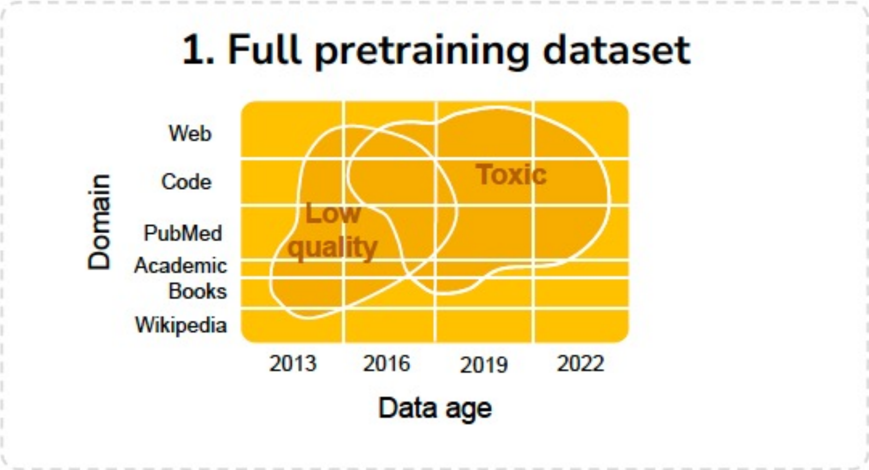
Source: Gopher paper

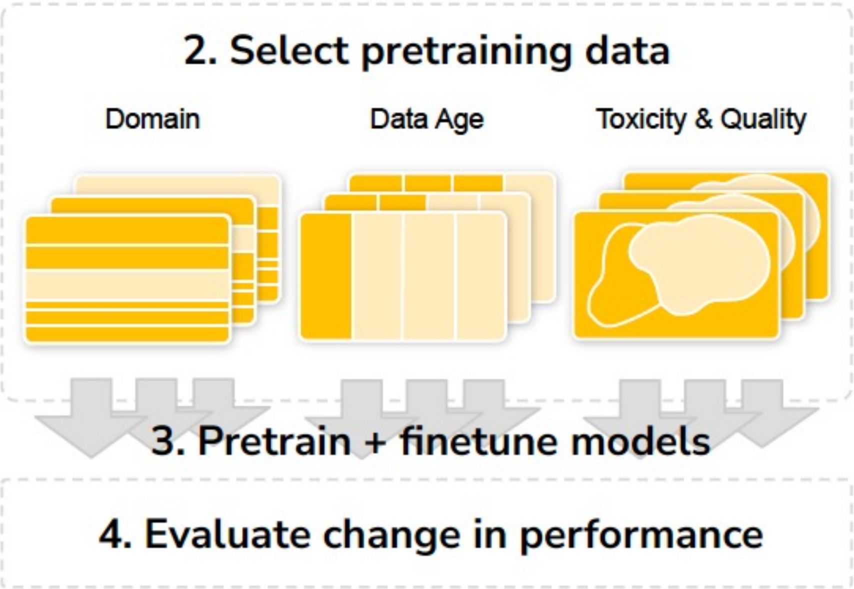
Common Practice in Pretraining Data Curation

Curation Decisions	*Frequently Disclosed*	*Guided by Intuition*	*Meaningful Impact*
Training data selection	❌	✅	✅
Scrape timestamp	~	✅	✅✅
Data cleaning	❌	✅	✅✅
Language filtering	~	✅	✅✅
PII removal	❌	❌	✅
Deduplication	❌	❌	✅✅
Toxicity / SafeURL filtering	❌	✅	✅
Quality filtering	❌	✅	✅
Sampling strategy	❌	~~✅~~ ❌	✅✅

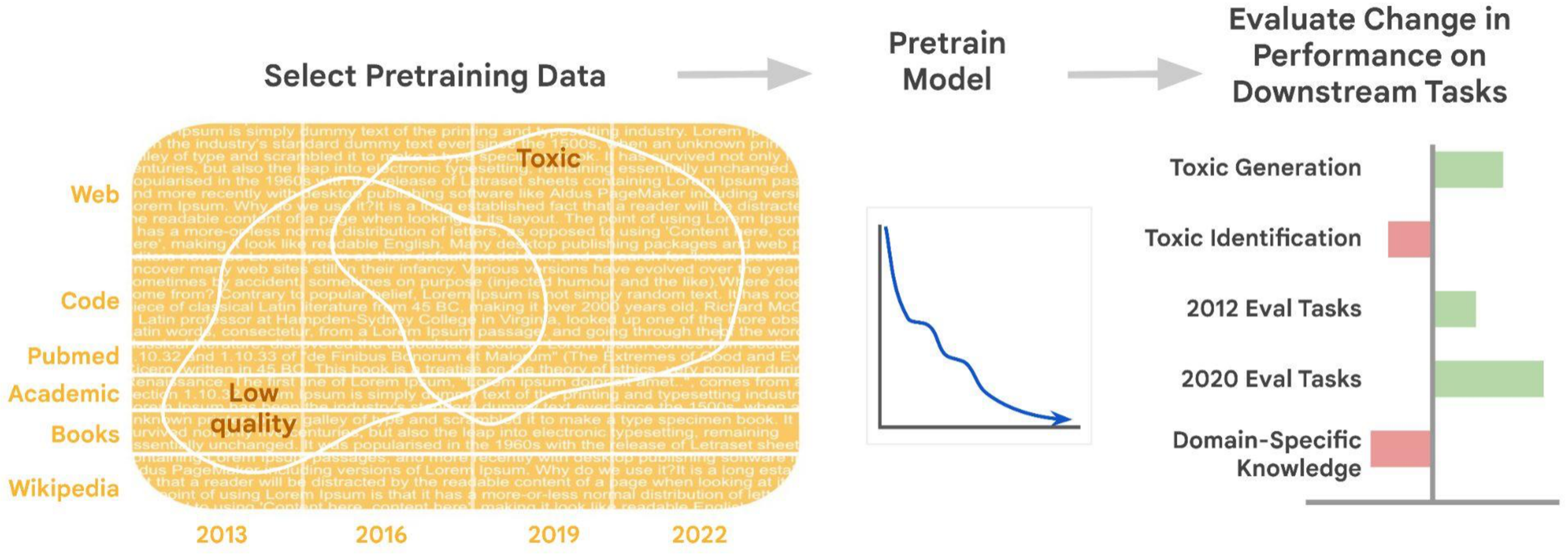
Research Objectives and the Approach of [Longpre et al, ACL2024]

- To evaluate how dataset design choices impact the final model
- Evaluate language models **pre-trained** on variants of training sets
    - Using C4 and Pile and their Variants as Datasets
- Use two language models (T5 variants)
    - LM-XL (1.5B)
    - LM-small (20M)





Summary of Approach



Data Curation Choices

Deduplication

Dataset Age

Dataset Domains

Quality Filters

Toxicity Filters

Focus of Study

- Effects of Data Age
- Effects of Quality Filter and Toxicity Filter
- Effects of Data Composition

Study the Effect of Dataset Age

Experiment Setup

- collected four variants of C4
- different snapshots of common-crawl with C4 recipe

Critique - CommonCrawl size has increased steadily over the years

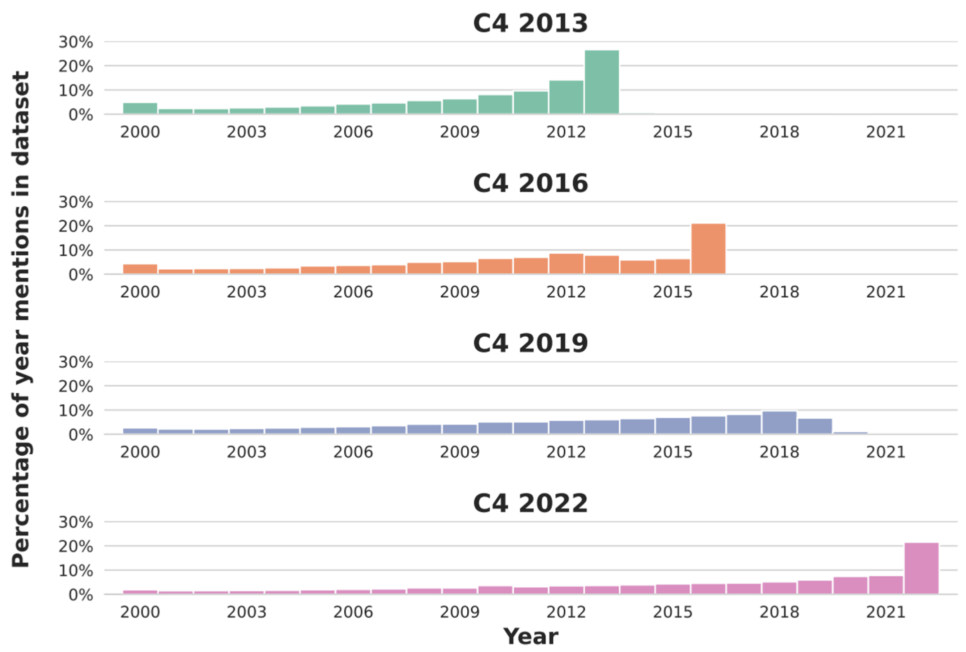


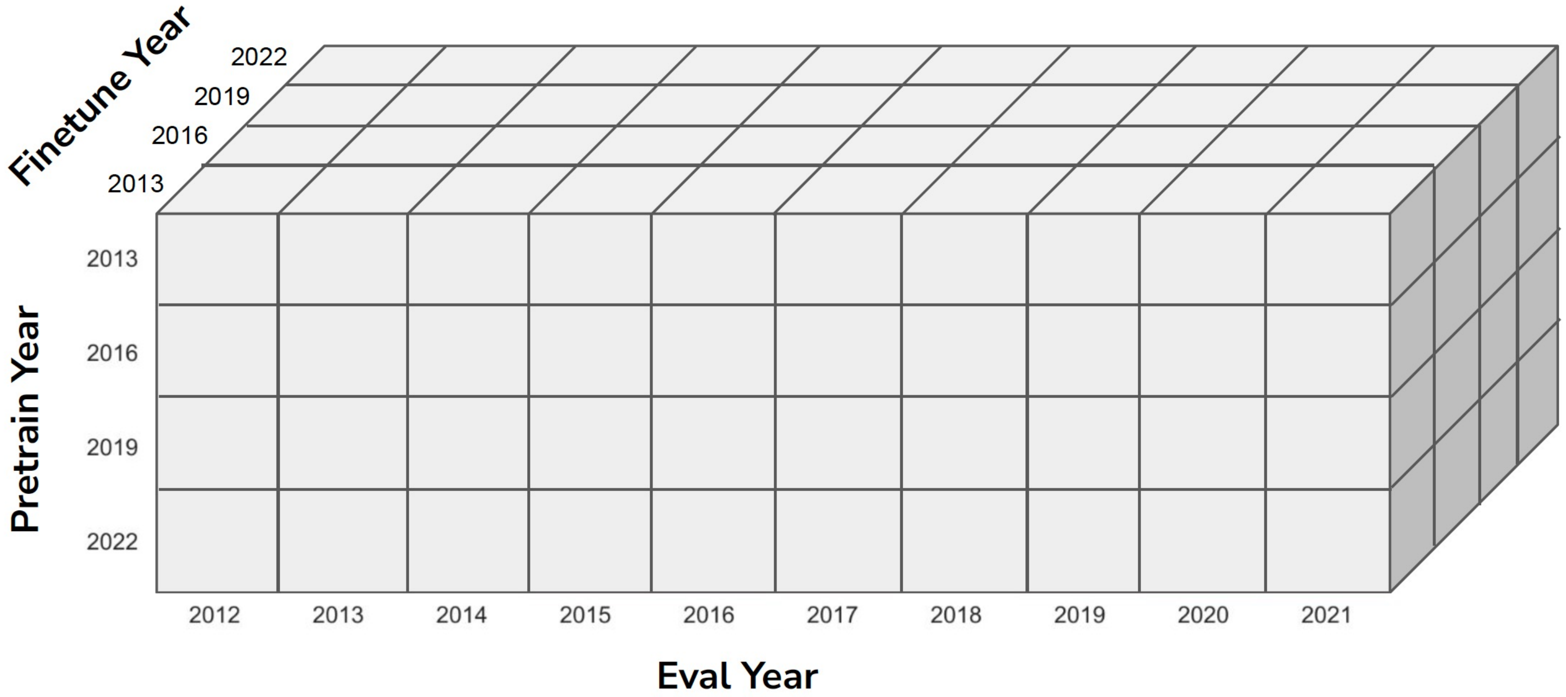
Figure 10: **Date instances in each of the C4 temporal pretraining versions.**

Study the Effect of Dataset Age

Experiment Setup

- construct four pre-trained models for each C4 version
- evaluate on tasks with test sets split by year
- measure temporal misalignment in performance

Data Age: Illustrative Dataset



A sample experiment on studying effects of Dataset Age

Evaluation - PubCLS Dataset

News source classification task

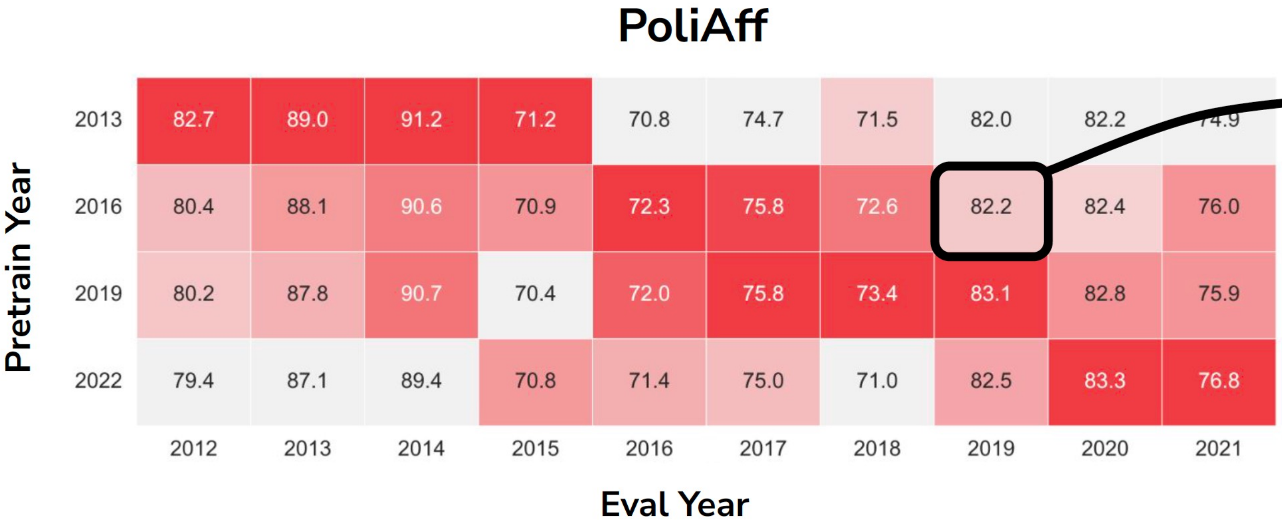
Accuracies for test set split over test split from different years

- pretraining data age somewhat correlated with evaluation metrics by age (0.61 pearson)

PubCLS

	2010	2012	2014	2016
2013	78.9	79.2	78.5	75.1
2016	76.8	78.7	79.0	76.3
2019	75.0	76.3	77.1	73.2
2022	74.0	75.7	76.8	73.4

Illustrative Results of Effect of Data Age



accuracy after pretraining on 2016 data, evaluating on 2019 data

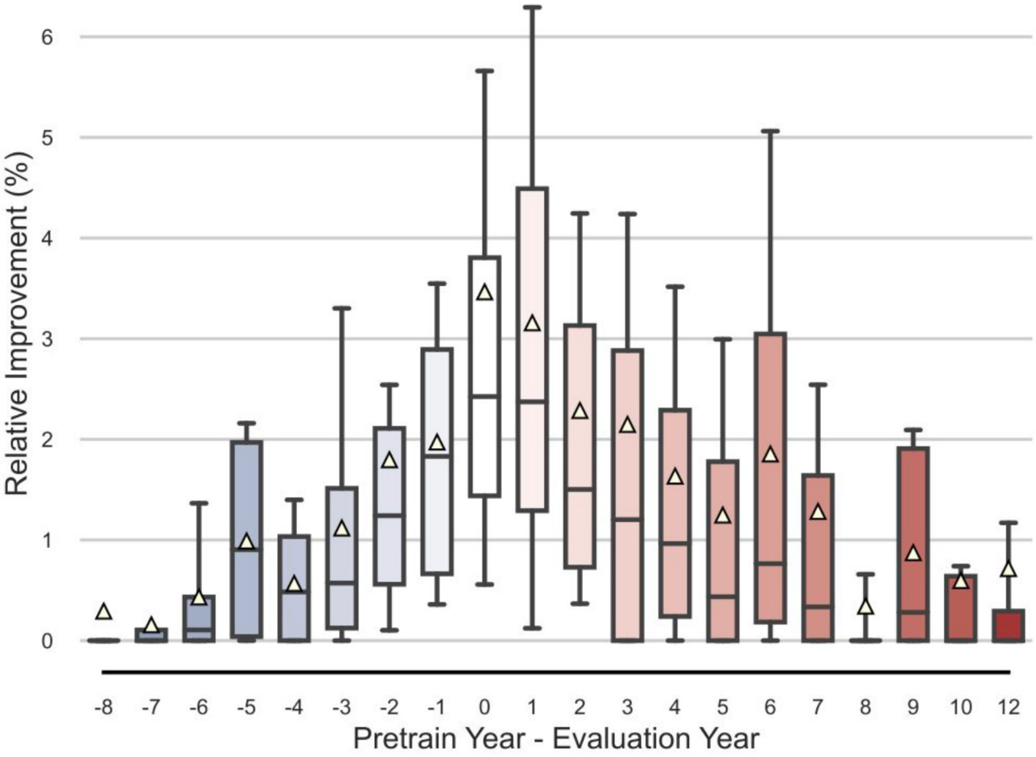
(averaged across finetuning years)

Observations on The Effect of Dataset Age

Data drift

- evaluating on “newer data” hurts
  - model trained on “older data” doesn’t know how to answer questions about covid
- evaluating on “older data” hurts
  - model trained on “newer data” doesn’t know how to answer questions about obama era!?
- another domain
  - github data starting from 2022 is proliferated with openai calls!

Key Findings of Data Age Effects



Takeaways:

1. Models and datasets become stale.
2. Temporal degradation persists even after finetuning.
3. Temporal degradation happens faster when evaluating old models on new benchmarks.
4. The effects of pretraining temporal misalignment are stronger for larger models than smaller models

Content Filtering based on Toxicity and Quality

Broad goals:

- Best downstream performance across tasks
- Prevent models from generating toxic text
- Identify toxic text

Quality filters in practice: Almost all models filter for some notion of quality

Toxicity filters in practice: T5, LaMDA, Chinchilla remove pretraining documents that might be toxic. Most models don't filter or don't disclose filtering.

Question: How does filtering pretraining documents based on toxicity and quality actually affect downstream tasks?

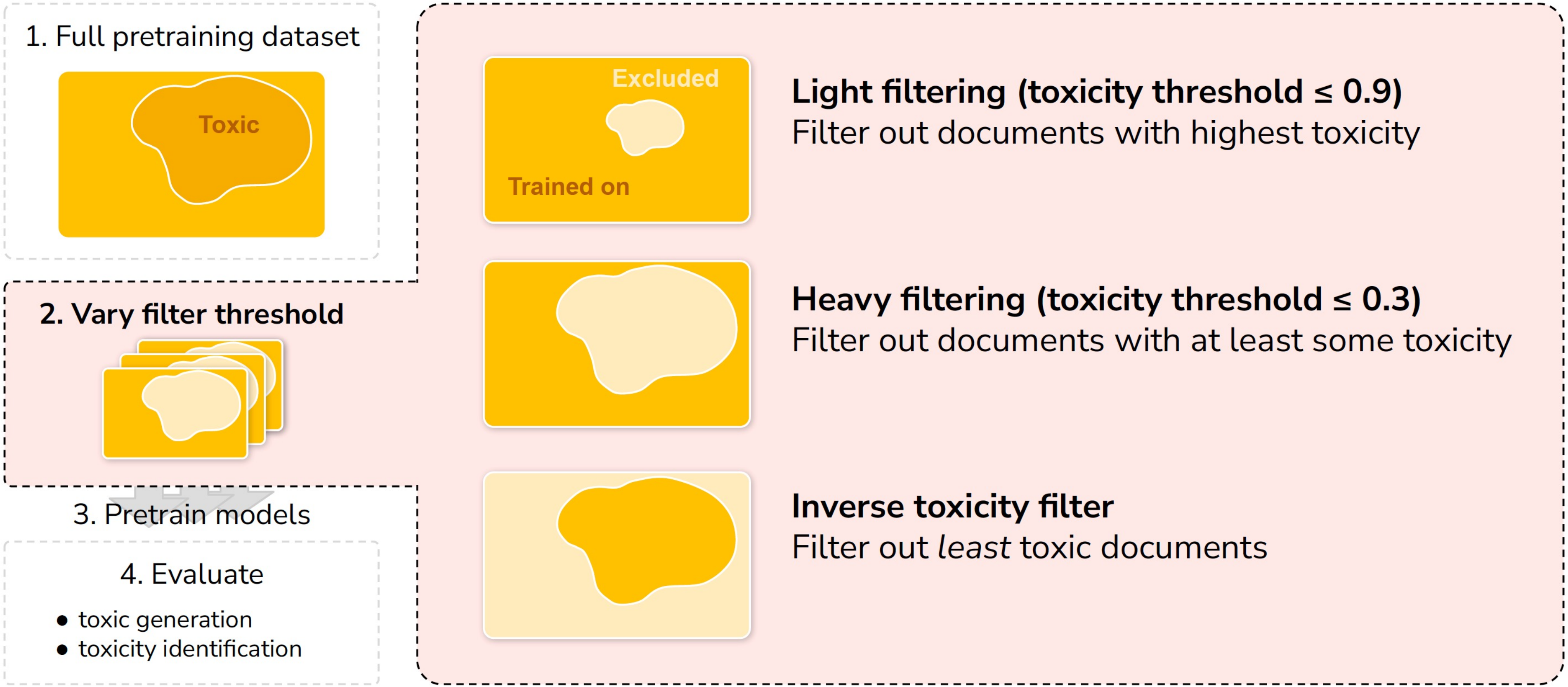
Background on Quality and Toxicity

- Modern LLM training workflows typically employ some form of quality and/or toxicity filtering
    - Quality heuristics are applied to web crawl data to filter out “low-quality” data
        - Newer models (e.g. GPT-3 and PaLM) now use quality classifiers
    - Toxic content is removed by applying heuristics or classifiers (e.g. SafeSearch filters)
- Definition
    - toxic = text that is profane, explicit, insulting, or threatening
    - quality = text similar to known “high-quality” sources

Quality and Toxicity Filtering Experiment Setup

- Quality filter:
  - Classifier employed by GLaM, PaLM, Chinchilla/Gopher
  - Assigns a score from 0 (Highest Quality) to 1 (Lowest Quality).
  - A “feature hash based linear classifier for inference speed”
  - Trained to classify between curated text (wiki, books + few select websites) and other.
- Toxic filter:
  - Jigsaw’s Perspective API
  - Trained on comments from online forums, labeled by annotators. Shown to be imperfect (reflects biases of annotators, false positives, etc), it has been shown to be far more accurate than rule-based classifiers.
  - Assigns a score from 0 (unlikely to be toxic) to 1 (very likely to be toxic).
- Implement toxic and quality filtering methods at different thresholds to vary the quantity of low-quality/toxic content present in C4/Pile → Analyze effect on downstream tasks.

How to create Toxic-Filtered Dataset ?



Considerations for Toxicity Filtering

1. Full pretraining dataset

2. Vary filter threshold

3. Pretrain models

4. Evaluate

- toxic generation
- toxicity identification

Toxic generation: Is generated text considered toxic?

Datasets:

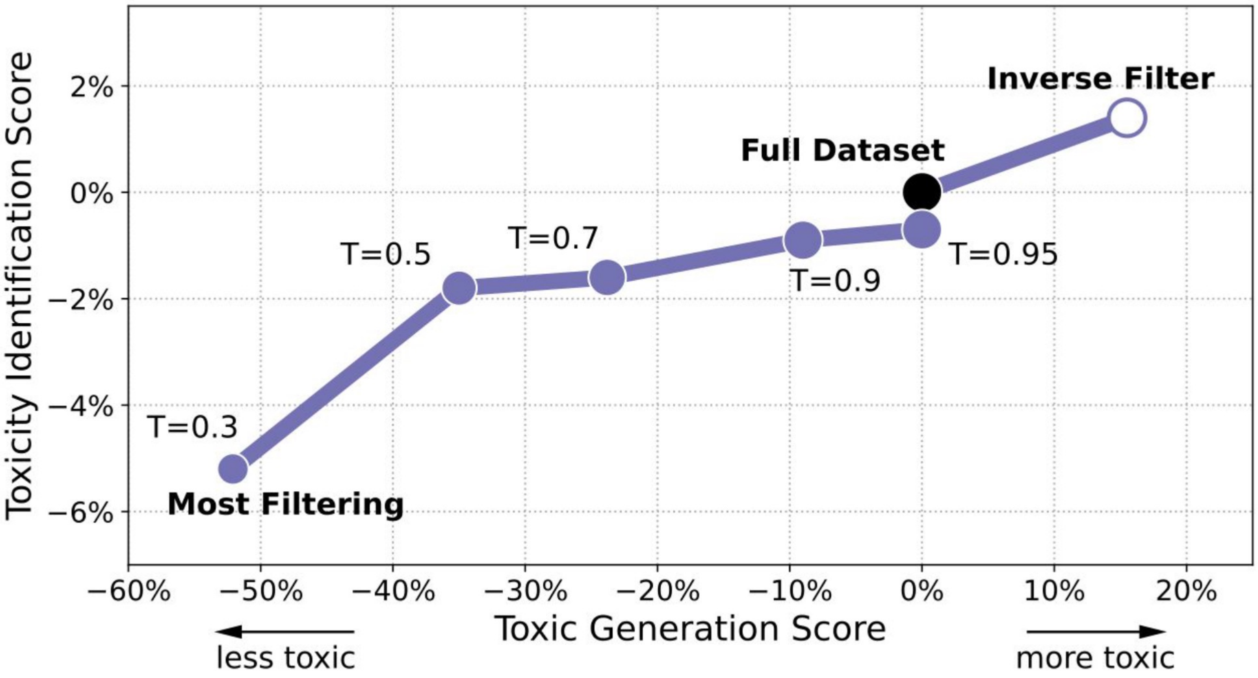
- RealToxicityPrompts (Gehman & al., 2020)
- RepBias (Chowdhery & al., 2022)

Toxicity identification: Can the model classify text as toxic?

Datasets:

- Social Bias Frames (Sap & al., 2020)
- DynaHate (Vidgen & al., 2021)
- Toxigen (Hartvigsen & al., 2022)

Toxicity: Tradeoff b/w Identification and Generation



Takeaways:

1. Toxicity filtering reduces toxic generation at the cost of decreased identification.

2. If the goal is to identify toxic text, then training on toxic data is more effective.

NB: Inverse Toxic Filter => Filter out Least Toxic documents

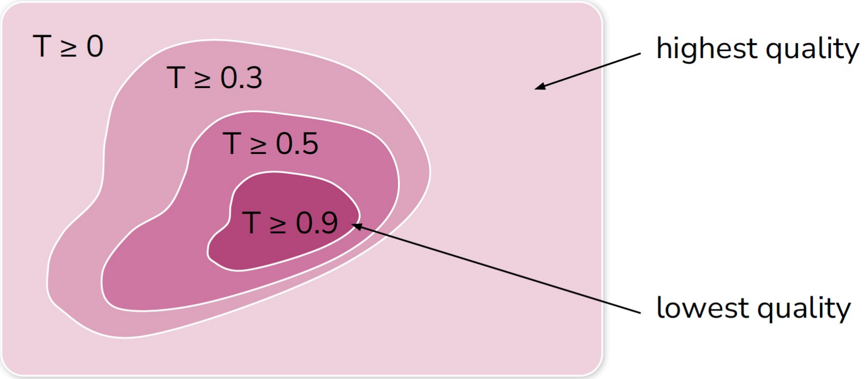
Quality Filtering's Effect on Toxicity Evals

- Same Setup, Baseline and Evals as Toxicity Filtering
  - EXCEPT Filter Pretraining Dataset by Quality instead of Toxicity
- But how to Measure "Quality" ?
  - An example:

GLaM/PaLM classifier:

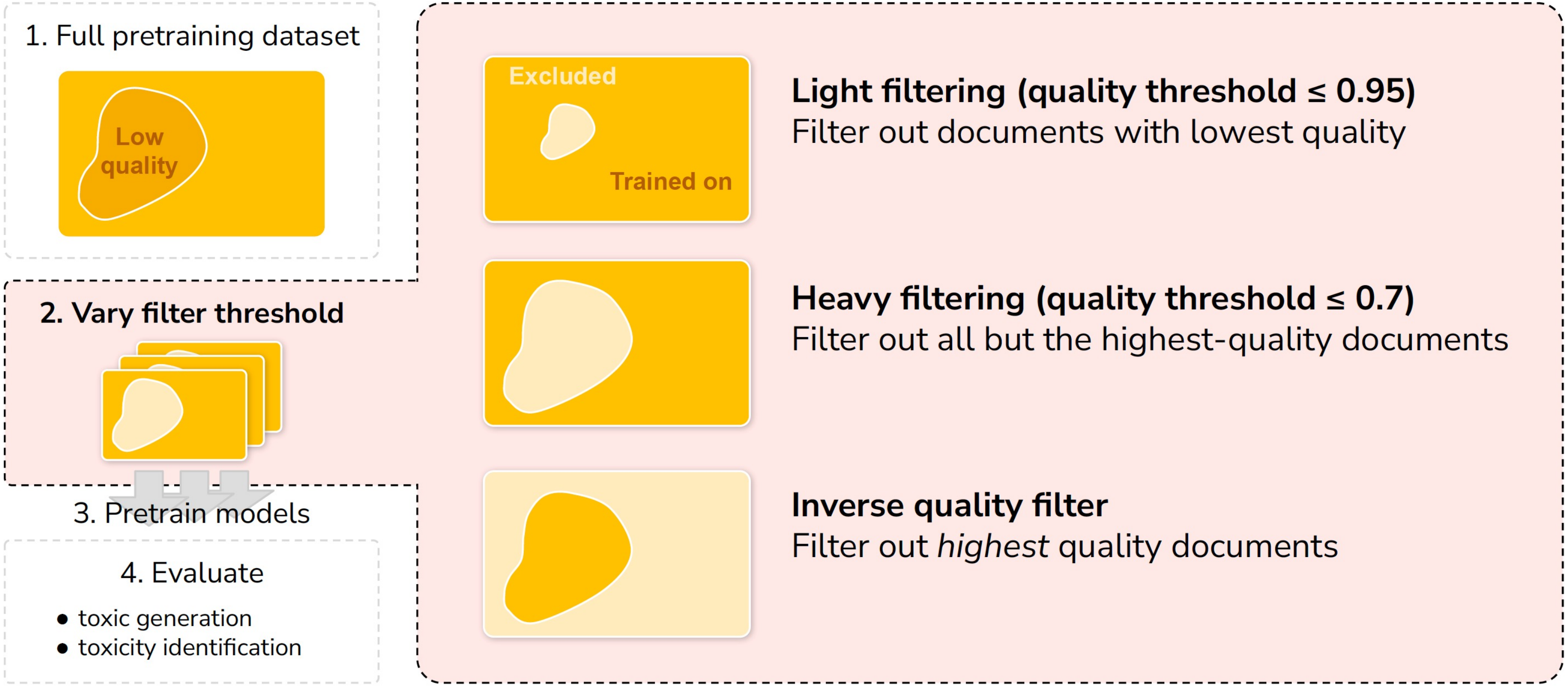
(Du & al., 2022)
(Chowdhery & al., 2022)
GPT-3, probably GPT-4

- Wikipedia + books are high quality
- every document gets a score from 0 (high quality) to 1 (low quality)



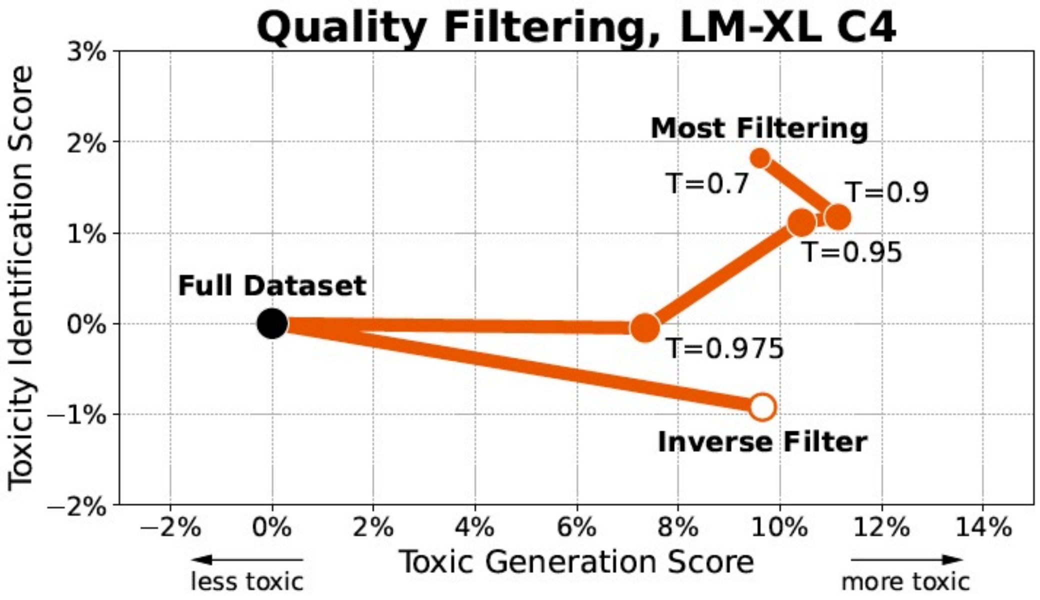
This is an existing operationalization, with many downsides.

Quality Filtering's Effect on Toxicity Evals



- Finding: Quality Filtering Improves Toxicity Identification

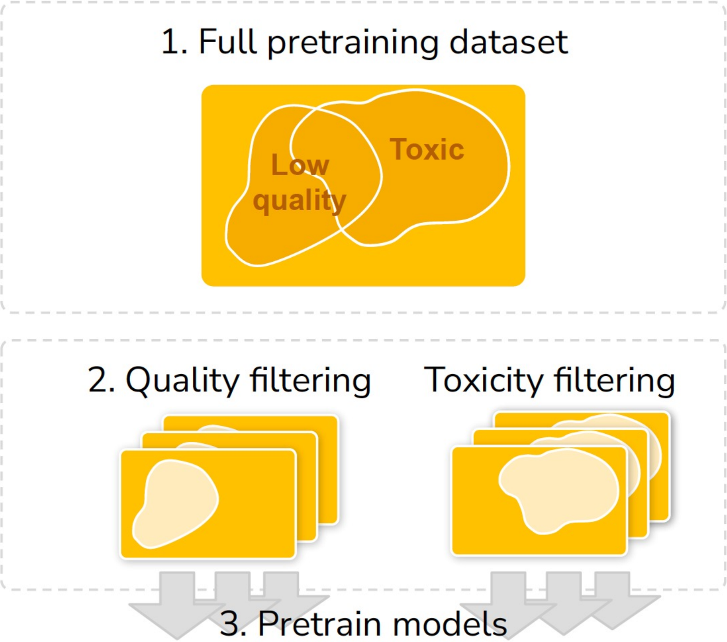
Impact of Quality Filters on Pretrained Models



NB: Inverse Quality Filter => Filter out Highest Quality documents

- Surprising Finding: Quality Filtering increases both capability of Toxic Generation and Toxic Identification !

Effect of Quality & Toxicity Filters on Downstream Task Performance



4. Evaluate

- QA tasks, grouped by domain

Question answering:

27 QA tasks from:

- MRQA (Fisch & al., 2019)
- UnifiedQA (Khashabi & al., 2020)

Categorized by domain:

- Wiki
- Web
- Academic
- Commonsense

Effect of Quality & Toxicity Filters on Downstream Task Performance

	Filter	Data	Wiki	Web	Acad	CS	Mean
			QA domain				
Baseline	Full Data	100%	0	0	0	0	0
Toxicity	Light (T=0.9)	95%	-2.2	-1.1	+0.2	+0.2	-0.7
	Heavy (T=0.5)	76%	-4.2	-2.4	-1.1	-3.5	-2.7
	Inverse	92%	+0.4	-1.4	+4.9	+2.7	+1.7
Quality	Light (T=0.975)	91%	+1.2	+0.7	+6.4	+6.1	+2.5
	Heavy (T=0.9)	73%	-0.3	+0.8	+0.8	+6.8	+1.2
	Inverse	73%	-5.0	-4.5	-2.7	-6.4	-3.1

Takeaways:

1. Toxicity filtering hurts performance across domains.
2. Quality filtering improves performance across most domains, despite removing data.

NB: Inverse Quality Filter => Filter out Highest Quality documents
Inverse Toxic Filter => Filter out Least Toxic documents

Impact of Quality & Toxicity Filters on Pretrained Models

Section Findings

- Quality and toxicity filters have very different effects.
- Quality filters improve performance significantly, despite removing training data.
- Quality filtering effects are not easily predicted by dataset characteristics. Future filters should weigh more than one dimension of quality.
- Toxicity filtering trades off generalization and toxicity identification ability for reduced risk of toxic generation.
- When optimizing for toxicity identification tasks, practitioners should use an inverse toxicity filter.

Recommendations for Quality and Toxicity Filtering

- If the Goal is to Identify Toxic Text, then don't use Toxicity Filters
- Use Quality Filters generally improves performance despite removing training data
- Should Investigate other kinds of Quality Filtering, not just Similarity to Books and Wikipedia

Goal of Dataset Domains/Composition Experiments

- Pile combines multiple different dataset domains
- Identify "High quality" domains would help model perform better

e.g. code data is often linked to "reasoning" capabilities of LLMs[1]

1. At Which Training Stage Does Code Data Help LLMs Reasoning?

Dataset Domains/Composition Experiment Setup

- Pile comprise of 22 different domains. Sub-group it into 9 high-level classes:
    - CommonCrawl, Web, Wikipedia, Books, Academic, Biomed, Legal, Code, Social/ Dialog
- Pre-train a model on dataset without one of the nice components

- Critique - the different components

- Dataset have different sizes
- Difference characteristics, relationships with test set
- Evaluating LLMs is challenging
- What if models with significantly larger capacity were used?

Effects of Dataset Domains/Composition on QA Tasks

	Wiki	Web	Books	Biomed	Academic	Common Sense	Contrast Sets	Average
Full Dataset (100%)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
No Social (99%)	-0.8	-3.7	2.6	0.1	3.5	-3.5	3.5	0.4
No Wiki (98%)	-1.3	-5.3	3.0	0.2	0.9	-4.4	7.2	-0.3
No Books (93%)	-3.5	-6.3	1.0	0.0	-1.6	-6.5	-4.4	-2.7
No OpenWeb (93%)	-2.0	-4.1	0.1	-1.0	0.6	-5.8	-2.9	-1.4
No Legal (91%)	-2.7	-2.9	3.8	0.4	0.8	-2.6	-0.4	-0.6
No Academic (87%)	-0.3	-2.5	0.3	-0.9	2.2	-1.1	4.3	0.2
No Pubmed (85%)	-0.3	-3.0	3.9	-5.8	-1.5	-5.9	3.9	-1.2
No Code (81%)	-0.5	-3.1	2.9	-1.2	1.2	-5.8	4.4	-0.1
No CC (73%)	-3.2	-6.2	-2.9	-4.6	-5.9	-8.0	-5.2	-4.8

Figure 8: **QA tasks are affected by removing domains when pretraining LM-XL.** Each row represents a model with one domain removed, the size of the remaining dataset is shown at the left in parentheses. Each column represents a set of QA evaluations from a domain. The FULL DATASET model represents the unfiltered Pile LM-XL, and all scores are relative to this Base model.

Observations:

- Quantity and Diversity both come into play ; Larger Components like CommonCrawl (CC) are likely to be diverse and thus correlated
- Removing Books & Common Crawl domains hurt downstream performance most.
- Targeted Data helps for Targeted Evaluation.

Recommendations:

- Train on as much data as possible; Quantity matters more than Domain Composition
- Prioritize Heterogeneous Data Sources

Impact of Data Curation on Data Characteristics

Section Findings

- The Pile's documents are on average longer, more readable and higher quality than documents in C4 but contain more personally identifiable information (PII).
- Books is an outlier domain, having the longest, most readable, most toxic, and most PII-filled documents, while also containing high-quality text.
- High toxicity and low quality documents have similarly high PII amounts but otherwise have very different average length and quality and toxicity levels.
- More recent web-scraped text is more diverse and less toxic but also lower quality.

Summary

Key Takeaways

- ➔ Data is largely undocumented & unknown. Practitioners are *guided by intuition.*
- ➔ Stale pretraining data matters & is not overcome by finetuning!
- ➔ Temporal misalignment effects grow with model size.
- ➔ "Quality" filters boost performance, even while reducing training data.
- ➔ Toxicity filters hurt. Inverse toxicity filters can help a lot for some tasks.
- ➔ Data heterogeneity and quantity matter most, especially web and books data.

Key Limitations

- "Quality" is ill-defined & deserves more attention.
- Compute is expensive! But so is dark data & documentation debt.
- Blackbox APIs have limitations.

Deduplication

based on the paper:
Katherine Lee et al, “Deduplicating Training Data Makes Language Models Better,” ACL 2022.

Deduplication

- “Deduplicating Training Data Makes Language Models Better”
- Timeline:
    - after C4
    - concurrent with GPT-3
    - Used by PALM, Gopher/Chinchilla
- Why is this important?
    - Efficiency: Can train on more high-quality tokens for same budget
    - Reduce overfitting by eliminating train-test leakage
    - + other benefits explored in Anthropic paper
- Challenges
    - How to scale to massive datasets
    - Naive implementation (e.g. exact match) doesn’t work for all types of data

Advantages

- Reduce rate of emitting memorized training data in unprompted setting
- Reduce train-test overlap → reduce over-estimation of model accuracy
- Increase efficiency: reduce train time in terms of time, $
- Does not hurt perplexity

Focus of the Dedup paper [Lee et al, ACL 2022]

- Focused on how duplicate text in train/validation impacts model perplexity and the extend of memorized content on generated text
- Not on downstream performance

Exact string matching, aka “naive” dedup

Small interspersed differences make exact duplicate matching less effective

Dataset	Example	Near-Duplicate Example
Wiki-40B	\n_START_ARTICLE_\nHum Award for Most Impactful Character \n_START_SECTION_\nWinners and nominees\n_START_PARAGRAPH_\nIn the list below, winners are listed first in the colored row, followed by the other nominees. [...]	\n_START_ARTICLE_\nHum Award for Best Actor in a Negative Role \n_START_SECTION_\nWinners and nominees\n_START_PARAGRAPH_\nIn the list below, winners are listed first in the colored row, followed by the other nominees. [...]
LM1B	I left for California in 1979 and tracked Cleveland 's changes on trips back to visit my sisters .	I left for California in 1979 , and tracked Cleveland 's changes on trips back to visit my sisters .
C4	Affordable and convenient holiday flights take off from your departure country, "Canada". From May 2019 to October 2019, Condor flights to your dream destination will be roughly 6 a week! Book your Halifax (YHZ) - Basel (BSL) flight now, and look forward to your "Switzerland" destination!	Affordable and convenient holiday flights take off from your departure country, "USA". From April 2019 to October 2019, Condor flights to your dream destination will be roughly 7 a week! Book your Maui Kahului (OGG) - Dubrovnik (DBV) flight now, and look forward to your "Croatia" destination!

Proposed Algorithms in [Lee et al, ACL 2022]

- Exact substring deduplication (ExactSubstr)
- Approximate matching with MinHash (NearDup)

Exact Substring Duplication (ExactSubstr)

- Idea: 2 examples are duplicates if they share a sufficiently long substring

- There exists a linear runtime implementation of exact substring matching that uses a Suffix Array:

1. Practical space-efficient suffix array construction algorithms (SACAs) exist that require worst-case time linear in string length;
    - SACAs exist that are even faster in practice, though with super linear worst case construction time requirements;

2. Suffix arrays allow the identification of duplicates in linear time
    - Hundreds of research papers on the construction and applications of suffix trees and suffix arrays. Refer to the survey on Suffix Array Construction algorithms by Puglisi, Smyth, Turpin [PST07]

N.B.: Suffix arrays have become the data structure of choice for many, if not all, of the string processing problems to which suffix tree methodology is applicable.

Suffix Array - Introduction

Definition 1. Given a text S of length n, the *suffix array* for S, often denoted *suftab*, is an array of integers of range 1 to n specifying the lexicographic ordering of the suffixes of the string S.

It will be convenient to assume that $S[n] = \$$, where \$ is smaller than any other letter.

That is, $\mathrm{suftab}[j] = i$ if and only if $S[i \mathinner{..} n]$ is the j-th suffix of S in ascending lexicographical order. We will write $S_i := S[i \mathinner{..} n]$.

We will assume that n fits into 4 bytes of memory. (That is, $n < 2^{32} = 4\,294\,967\,296$.) Then the basic form of a suffix array needs only $4n$ bytes.

The suffix array can be computed by sorting the suffixes, as illustrated in the following example.

Source: https://www.mi.fu-berlin.de/wiki/pub/ABI/RnaSeqP4/suffix-array.pdf

Suffix Array – An Example:

The text is $S = abaababbabbb\$$, $n = 13$. The suffix array is:

Suffixes		Ordered suffixes		
i	S_i	i	suftab[i]	$S_{\text{suftab}[i]}$
1	abaababbabbb$	1	13	$
2	baababbabbb$	2	3	aababbabbb$
3	aababbabbb$	3	1	abaababbabbb$
4	ababbabbb$	4	4	ababbabbb$
5	babbabbb$	5	6	abbabbb$
6	abbabbb$	6	9	abbb$
7	bbabbb$	7	12	b$
8	babbb$	8	2	baababbabbb$
9	abbb$	9	5	babbabbb$
10	bbb$	10	8	babbb$
11	bb$	11	11	bb$
12	b$	12	7	bbabbb$
13	$	13	10	bbb$

It is tempting to confuse $suftab[i]$ with $S_{suftab[i]}$ since there is a one-to-one correspondence, but of course the two are completely different concepts.

Suffix Array

- Suffix array for sequence S is a lexicographically-ordered list of all suffixes contained in a sequence: A(S) = argsort(all_suffixes(S))
- 10-100x more memory efficient than suffix tree
- Procedure:
  - Concat entire dataset into sequence S
  - Construct A
  - Linearly scan A from beginning to end looking for sequences A_i, A_i+1 that share a common prefix of at least some threshold length
  - Easy to parallelize

Parallelized Implementation

1. Parallel partial suffix array construction
    a. O(N) work, O(N/K) wall-clock.
2. Parallel merge of partial suffix arrays
    a. O(N m log(K)) - m = average length of prefix match
3. Computational Analysis
    a. 96 cores, 768GB of memory
    b. 350GB C4 takes under 12 hours wall clock to build suffix array, 1 hour to dedup.
    c. Suffix array for 350GB has 8x overhead (1.5TB)

Approximate Matching (NearDup)

- Uses MinHash, an approximate matching algorithm widely used in dedup task
- Represent documents by a set of n-grams, then use hash functions to approximate the Jaccard Index
- Jaccard Index (JI) = (size of intersection / size of union)
    - 0 when sets are disjoint, 1 when equal, in [0, 1] when otherwise
- If Jaccard index is sufficiently high, documents are considered approximately matches of each other
- To efficiently approximate JI, MinHash constructs document signatures by sorting the n-grams via hash functions and keeping k smallest.

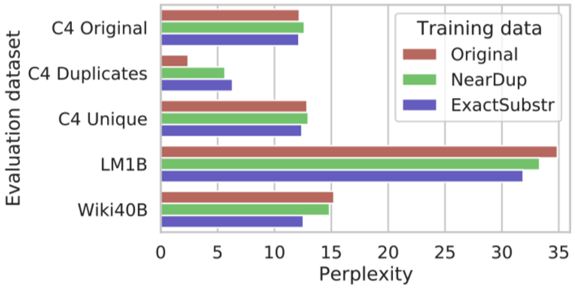
For details, refer to ”Locality Sensitive Hashing (LSH)” in IERG4300 Lecture Notes or the "MMDS” textbook by Leskovec, Rajaraman and Ullman

Results

1. Dedup results: 3% to 14% near duplicates on C4 and RealNews, Wiki < 1% near dups
2. Leakage: 4.6% of the C4 validation set and 14.4% of the RealNews validation set examples had an approximate duplicate in their respective training sets

Model	1 Epoch	2 Epochs
XL-ORIGINAL	1.926%	1.571%
XL-NEARDUP	0.189%	0.264%
XL-EXACTSUBSTR	0.138%	0.168%

Table 4: When generating 100k sequences with no prompting, over 1% of the tokens emitted from a model trained on the original dataset are part of a 50-token long sequence copied directly from the training dataset. This drops to 0.1% for the deduplicated datasets.



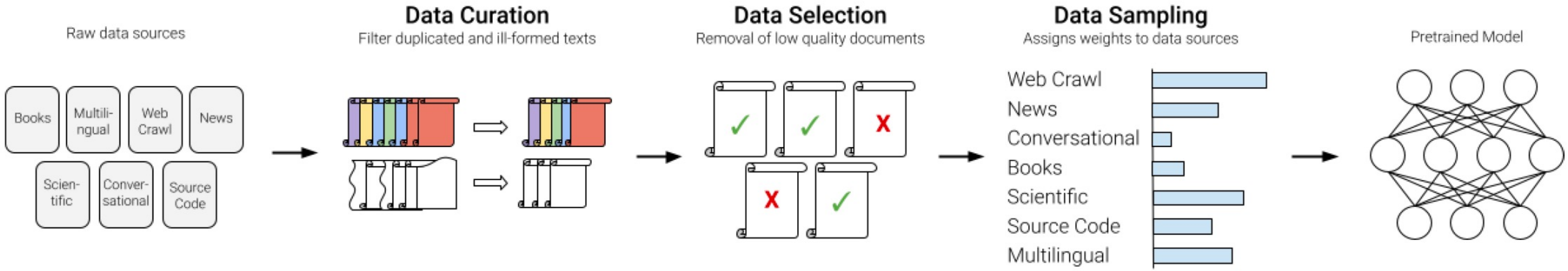
Data, Data Everywhere:
A Guide for Pretraining Dataset Construction

Jupinder Parmar*, Shrimai Prabhumoye, Joseph Jennings,
Bo Liu, Aastha Jhunjhunwala, ZhilinWang, Mostofa Patwary,
Mohammad Shoeybi , Bryan Catanzaro
of NVIDIA

EMNLP 2024

Naman and Kevin

Pretraining Data Processing Pipeline under Study



Data Sources used in this Study [Parmar et al EMNLP 2024]

Data type	Data source	Tokens (B)
English	Web crawl	889
	Misc	109
	News	94
	Conversational	59
	Books	35
	Scientific	33
Multilingual	Web crawl	540
	Parallel corpora	56
Source Code	The Stack v1.2	212

Data source	Dataset name	Tokens (B)
Web Crawl	CC 2022-40	284.3
	Re-crawled C4	174.8
	CC 2019-35	165.1
	CC 2020-50	141.9
	CC 2021-04	68.2
	Pile-CC	41.2
	OpenWebText2	14.0
News	CC NEWS	94.2
Misc	ROOTS	104.5
	Wikipedia	4.3
Conv.	Reddit + others	59.1
Books	Books3	25.1
	Stories	5.3
	Gutenberg	2.5
	BookCorpus2	1.5
Scientific	ArXiv	18.7
	StackExchange	9.8
	PubMed Abstracts	4.2
	NIH ExPorter	0.3

Table 11: Summary of each of the datasets that make up our English corpus

Data Sources used in Ablation Study

ISO	Tokens (B)	ISO	Tokens (B)	ISO	Tokens (B)	ISO	Tokens (B)
RU	94.52	FA	6.59	HI	2.60	IS	0.38
JA	70.52	RO	6.58	SK	2.58	UR	0.37
DE	48.98	TR	6.46	HR	2.45	AZ	0.37
ES	46.50	EL	6.43	CA	2.12	MR	0.33
FR	44.30	SV	6.39	LT	1.69	KA	0.32
ZH	43.41	HU	5.89	HE	1.47	MK	0.32
IT	26.40	AR	5.74	SL	1.33	NE	0.31
NL	15.64	NO	5.61	SR	1.24	KK	0.30
VI	15.16	FI	4.11	ET	1.24	HY	0.29
PL	14.50	DA	3.79	BN	0.90	GL	0.29
PT	11.99	UK	3.63	LV	0.84	ML	0.25
ID	10.90	BG	3.37	TA	0.82	TE	0.24
CS	7.23	KO	3.05	SQ	0.49	KN	0.18

Table 12: Summary of our multilingual web crawl data consisting of 52 languages. All languages except for JA and ZH were curated from the 2022-40 CC snapshot. The JA and ZH data were curated from the mC4 corpus.

Data Sources used in Ablation Study

Language	Tokens (B)	Language	Tokens (B)	Language	Tokens (B)
Javascript	21.12	Rust	2.81	Pascal	0.68
Markdown	20.27	Jupyter	2.58	Assembly	0.67
Java	19.84	Ruby	2.29	Fortran	0.65
Python	19.49	Swift	2.02	Makefile	0.54
PHP	18.87	JSON	1.78	Julia	0.52
C	18.26	$\TeX$	1.76	Mathematica	0.51
C++	15.79	Scala	1.29	Visual Basic	0.42
C#	12.05	YAML	1.28	VHDL	0.42
Go	9.03	Shell	1.18	Common Lisp	0.24
HTML	8.97	Dart	1.08	Cuda	0.21
Typescript	8.16	Lua	1.00	System Verilog	0.16
SQL	5.31	reStructuredText	0.96	Docker	0.16
CSS	4.96	Perl	0.83	Omniverse	0.03
XML	2.97	Haskell	0.72		

Table 14: Summary of our source code corpus consisting of 41 different programming languages all of which, except for omniverse, were curated from the Stack v1.2 dataset.

Operation Threshold Settings for Heuristics

Heuristic	Threshold	English Only
N-gram LM Perplexity	5000	Yes
Fraction of non-alpha-numeric characters	0.25	Yes
Fraction of words without alphabets	0.20	Yes
Fraction of numbers (in characters)	0.15	
Fraction of URLs (in characters)	0.20	
Fraction of lines starting with bullets	0.90	
Fraction of whitespaces (in characters)	0.25	
Fraction of parentheses (in characters)	0.10	
The ratio of symbols to words	0.10	
Contains a word >1000 characters	1.0 (Hard Constraint)	
Contains <50 or >100k words	1.0 (Hard Constraint)	
Contains less than 2 common English words	1.0 (Hard Constraint)	Yes
Mean word length <3 or >10 characters	1.0 (Hard Constraint)	
Fraction of boilerplate content (in characters)	0.40	
Duplicate line fraction	0.30	
Duplicate paragraph fraction	0.30	
Duplicate lines (by character fraction)	0.20	
Duplicate paragraph (by character fraction)	0.10	
Repeating top n-gram fraction	0.20	
Repeating duplicate n-gram fraction	0.20	
Fraction of lines that do not end with punctuation	0.85	
Fraction of lines that end with ellipsis	0.30	
Documents containing Pornographic content in URLs	1.00	

Table 15: A list of document-level data filtering heuristics and thresholds. Heuristics are borrowed or derived from Rae et al. (2021) and C4's cleaning heuristics (Raffel et al., 2020)

Operation Threshold Settings for Heuristics

Heuristic	Min. Threshold	Max Threshold
Fraction of comments (in characters)	0.001	0.85
Number of lines of code	5	20,000
Ratio of characters to tokens	2	-

Table 16: A list of file-level data filtering heuristics and thresholds applied to the source code data. Heuristics follow those described in (Allal et al., 2023).

Effects of Data Curation

Findings

- Compared to raw text, deduplicated and quality filtered data improve model accuracy.
- In deduplication, it is better to priortize keeping samples from older sources than more recent ones.

Experiment	LM-Eval
Raw text	57.18
Post deduplication	58.93
Post quality filtering	**59.50**

Table 2: Impact of data curation steps on model accuracy. Per-task accuracies are shared in Table 18.

Experiment	LM-Eval
Random	59.96
Recent-to-Old	58.93
Old-to-Recent	**60.47**

Table 3: The priortization of data sources in deduplication affects model accuracy. Per-task accuracies are shared in Table 19.

Effects of Data Selection using Domain Selection via Importance Sampling (DSIR) [Xie et al, NeurIPS 2023]

Findings

- DSIR improves the quality of web crawl snapshots.
- DSIR functions best when applied across each data source individually.
- DSIR is fairly sensitive to the composition of the target distribution.

Q1: How does naïve application w/ recommended settings of DSIR perform ?
Q2: Can we identify better settings for DSIR ?

Question	Experiment	LM-Eval
Q1	Original CC	54.30
	DSIR	**54.44**
Q2.1	Corpus DSIR	54.44
	Source DSIR	**54.71**
Q2.2	DSIR (80%)	54.55
	DSIR (87.5%)	54.25
	DSIR (95%)	**54.71**

Table 4: DSIR improves the quality of web crawl data. () refers to the percentage of examples that are selected by DSIR. Per-task accuracies are shared in Table 20.

Target Set	LM-Eval
Wikipedia, Books	**54.71**
Wikipedia, Books, arXiv, NIH	54.02
arXiv, NIH	53.71

Table 5: DSIR is impacted by target set composition. Per-task accuracies are shared in Table 21.

Note: DSIR [Xie et al, NeurIPS 2023] takes as input a raw dataset, along with a target dataset of known high quality examples, and then uses importance resampling to select examples from the raw dataset that are distributed like the target by utilizing a bag of hashed n-gram models to match the n-gram frequencies of the selected data and the target.

Effects of Data Sampling Methodologies: UniMax vs. Alpha Sampling vs. DoReMi

Findings

- UniMax provides the best sampling weights for the English and multilingual domains.
- Alpha sampling, with a value of $\alpha = 1.3$, provides the best sampling weights for the code domain.
- DoReMi is unable to produce competitive sampling weights for any domain as it often gives the majority of the weight to a single source.

Method	LM-Eval	MMLU
Preference	65.85	27.20
UniMax (1e)	**67.14**	**28.30**
UniMax (2e)	66.50	28.00
UniMax (4e)	66.61	26.60
DoReMi	65.63	26.90

Table 6: UniMax sampling weights provide the best performance on English data. Ne means that UniMax can use a maximum of N epochs per dataset. Per-task accuracies are shared in in Table 22.

Method	MultiPL-E	HumanEval
Alpha ($\alpha = 1.3$)	**19.72**	**20.73**
UniMax (1e)	19.33	20.12

Table 8: Alpha sampling outperforms UniMax on code data. Per-language accuracies for MultiPL-E are shared in Table 23.

Method	XCOPA	TyDiQA-GoldP
Alpha ($\alpha = 1.3$)	58.11	17.86
UniMax (1e)	**58.24**	**18.11**
DoReMi	57.65	15.8

Table 7: UniMax slightly outperforms alpha sampling on multilingual data.

Attribute Analysis

Findings

- Website homepages, news articles, and blogs constitute the majority of web crawl documents. Conversational texts are sparsely contained.
- Technical domains like finance, law, and science are among the least represented in web crawl.
- Explanatory or news articles on science and health are the most likely to be high quality documents.
- Domains or types of speech that are generally of high quality may also exhibit high toxicity (i.e news articles on sensitive topics), explaining why previous toxicity based filtering has harmed model accuracy.

Attribute Analysis (cont'd)

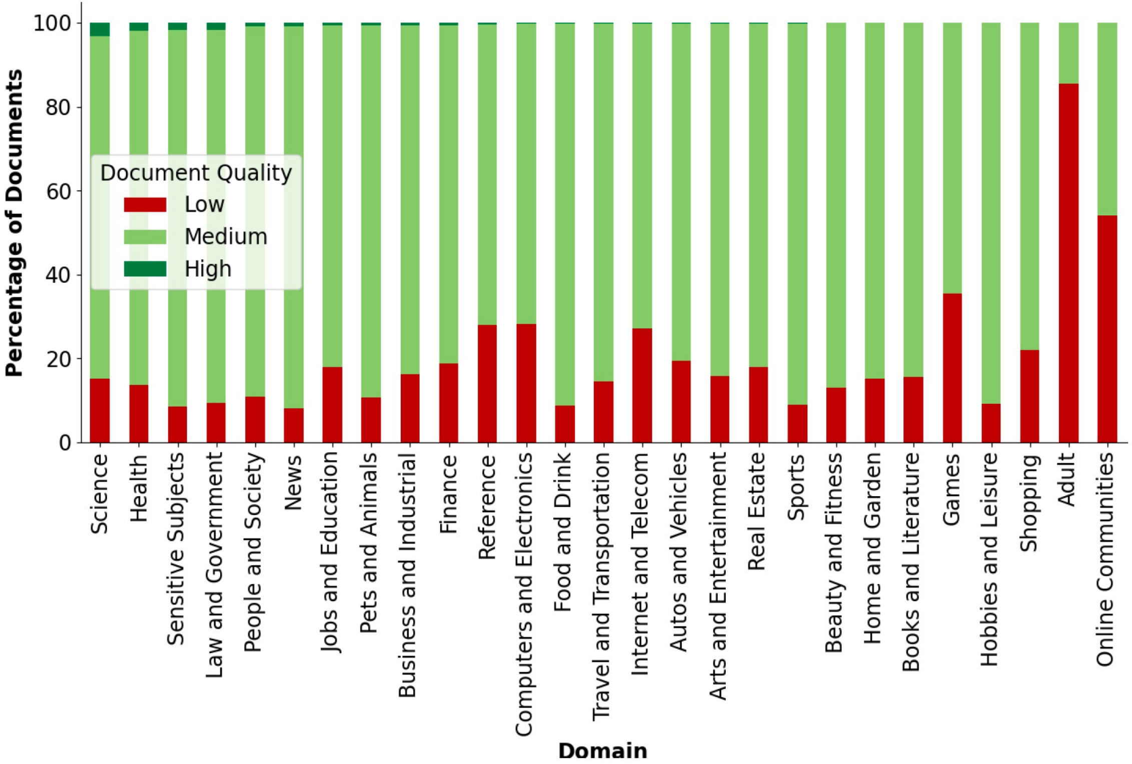


Figure 4: Domains sorted by descending order of percentage of high quality documents.

Attribute Analysis (cont'd)

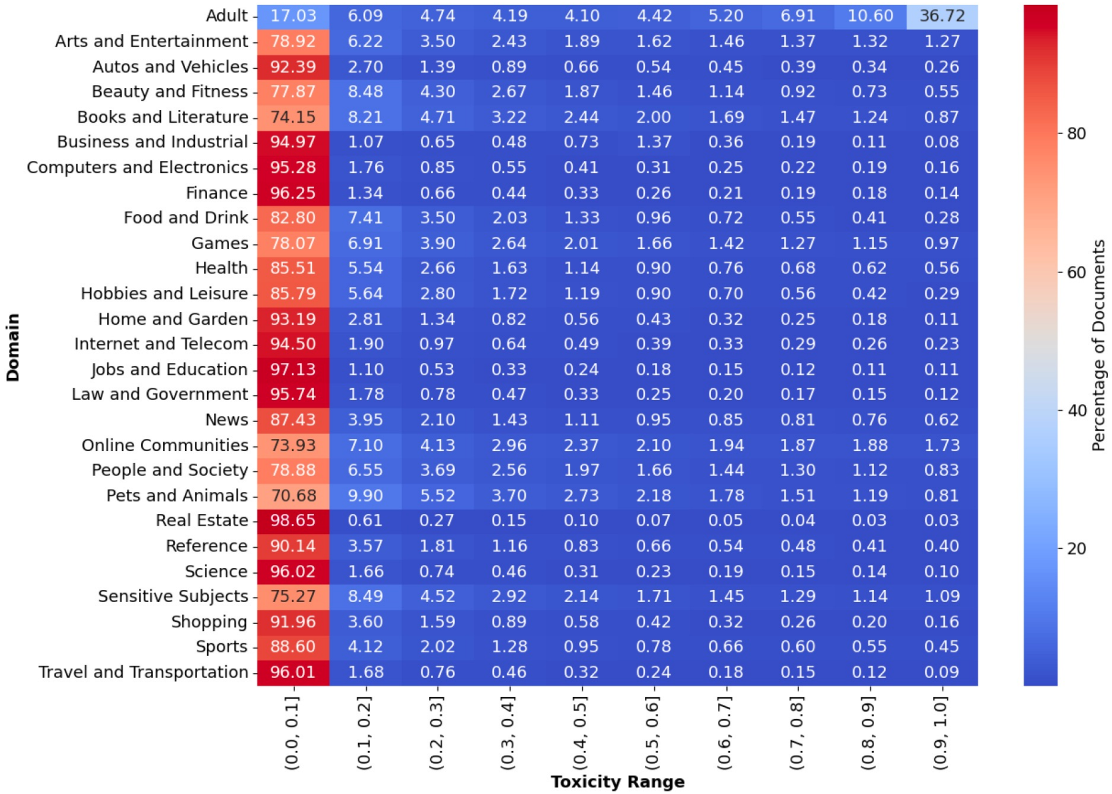


Figure 5: Heatmap of domains by probability of toxic content. Adult and online communities contain the highest percentage of toxic content.

Attribute Analysis (cont'd)

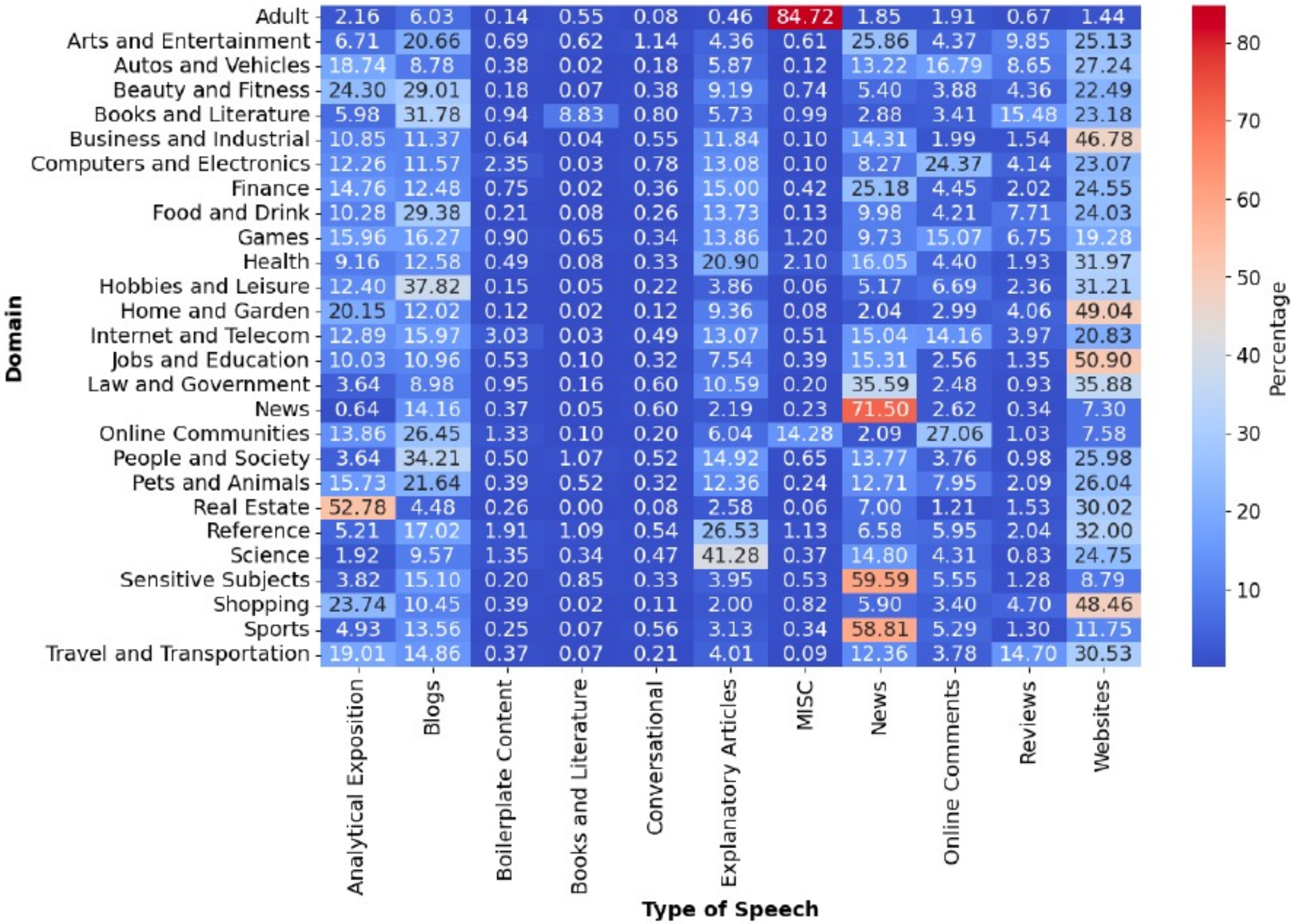


Figure 14: Heatmap of domains by types of speech.

Attributes in Sampling and Selection

Findings

- Buckets defined by data attributes substantially improve the performance of data sampling methods.
- Attributes compose more useful target sets for data selection.

Experiment	Target Set	LM-Eval
Original CC	N/A	54.90
DSIR	Wikipedia, Books	55.35
DSIR	Low Tox, High Qual	**55.63**

Table 10: Attribute information defines better target sets for data selection. Tox is Toxicity, Qual is Quality.

Experiment	LM-Eval
Baseline	56.81
Quality `fine-grained`	***57.88***
Quality `grouped`	*57.19*
Toxicity `fine-grained`	53.62
Toxicity `grouped`	54.99
Domain `fine-grained`	*57.34*
Domain `grouped`	*57.45*
Type of Speech `fine-grained`	56.69
Type of Speech `grouped`	*57.31*

Table 9: Sampling weights based on buckets of data attribute labels significantly improve upon baseline results. Italics indicate results that outperform the baseline. Per-task accuracies are shared in Table 24.

Data Curation Ablations

Experiment	HumanEval	MultiPL-E
Raw source code	16.5	15.9
Post quality filtering	20.7	19.2

Table 17: Evaluation accuracies before and after data curation for our source code dataset. We train an 8B model for 150B tokens.

The State of Data Curation at NeurIPS: An Assessment of Dataset Development Practices in the Datasets & Benchmarks Track

UNIVERSITY OF TORONTO
FACULTY OF INFORMATION
Digital Curation Institute
Just Sustainability Design Lab

Eshta Bhardwaj, Harshit Gujral, Siyi Wu, Ciara Zogheib, Tegan Maharaj, & Christoph Becker

NEURAL INFORMATION PROCESSING SYSTEMS

INTRODUCTION

Introduction

- NeurIPS has responded to the rising urgency and recognized impact of data research through the introduction of the **D&B track**
- This track aims to address the issue of datasets being used **outside their original scope**

Background

- Data curation involves "maintaining and adding value to digital research data for current and future use"
- **Field** of data curation has **established methods and discourse** on how to maintain large amounts of data and manage ethical concerns

Motivation

- ML research has turned towards the **improvement of data to improve model results** and fundamental understanding
- Current **gap in recognition and uptake** of data curation concepts in the ML community

Our Goal

- Document and improve the standard of **dataset development in NeurIPS** so that future benchmarks and datasets can be effectively found, easily accessed, ethically used, consistently evaluated, and appropriately reused

RESEARCH QUESTIONS & METHODS

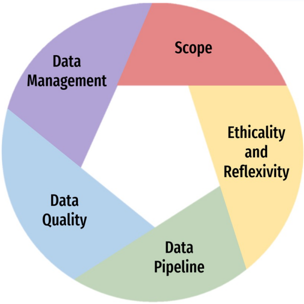
What constitutes a well curated dataset?

- Developed an **evaluation framework** made up rubric and toolkit
- **Rubric** evaluates dataset contents and dataset design decisions
- **Toolkit** provides application guidance for the rubric

What is the state of data curation at NeurIPS?

- Applied framework to **evaluate** NeurIPS datasets
- Examined the consistency in application by measuring **inter-rater reliability** (IRR)
- Assessed datasets to **evaluate current practices of data curation** in ML dataset development
- Analyzed areas in which **improvement** was needed

EVALUATION FRAMEWORK



- Context, purpose, motivation; Requirements
- Ethicality; Domain knowledge & data practices; Context awareness; Environmental footprint
- Data collection; Data processing; Data annotation
- Suitability; Representativeness; Authenticity; Reliability; Structured documentation
- Findability; Accessibility; Interoperability; Reusability

Context awareness

Context awareness demonstrates an understanding of the subjective, non-neutral nature, and situatedness of data.

Criteria to meet minimum standard

Documentation includes a positionality statement.

Criteria to meet standard of excellence

Documentation adopts a reflexive approach to dataset development. For example, documentation discusses how field epistemologies impact assumptions, methods, or framings.

STRATEGIES TO IMPROVE DATA CURATION IN ML

Requirements
- Create **purpose statements**
- Document **initial formulation** vs. the dataset creation scheme

Ethicality
- Consider **proportionality principle**
- Reflect on whether and how benefits outweigh the harms

Context awareness
- Include **positionality statements** to increase reflexivity on how identity impacts data-related choices

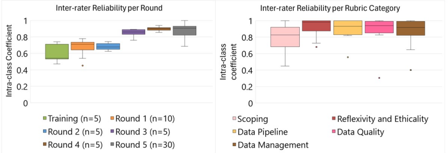
Environmental Footprint
- **Quantify** the environmental footprint of datasets to improve transparent reporting of resource consumption

Findability
- Assign **persistent identifiers** to metadata and host in a searchable repository

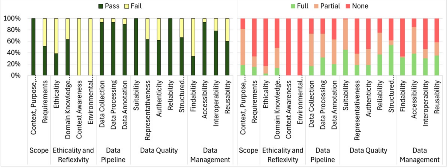
Reusability
- Include **identifier** information, dataset **characteristics**, and dataset **provenance**

CURRENT PRACTICES OF DATA CURATION

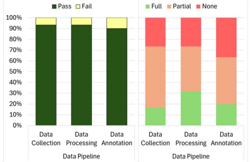
Finding 1 Inter-rater reliability suggests the evaluations are consistent and reliable



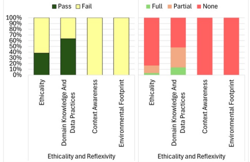
Finding 2 Documentation often remains incomplete



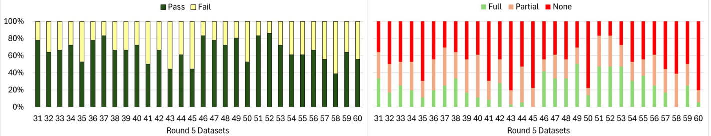
Finding 3 NeurIPS prioritizes model-work adjacent documentation



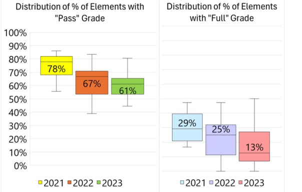
Finding 4 Documentation is rarely context-aware and typically does not quantify environmental footprint



Finding 5 Documentation quality varies widely across datasets



Finding 6 Findings suggest no improvements occurred over time



KEY TAKEAWAY

The creation of the D&B track shows that **dataset quality is the foundation of continued progress in ML** applications. There is no better database of knowledge than data curation to aid in this venture.

Our evaluation framework provides a practical lens on how NeurIPS can spearhead the requirement for **rigorous data curation in ML.**

A recent work from NeurIPS 2024 to "improve dataset development for benchmarking and datasets".

Tokenization

Tokenization

What is Tokenization ?

- Tokenization is the process of **breaking down a piece of text**, like a sentence or a paragraph, into individual words or “tokens.”
- These tokens are the **basic building blocks of language**, and tokenization helps computers understand and process human language by splitting it into manageable units.

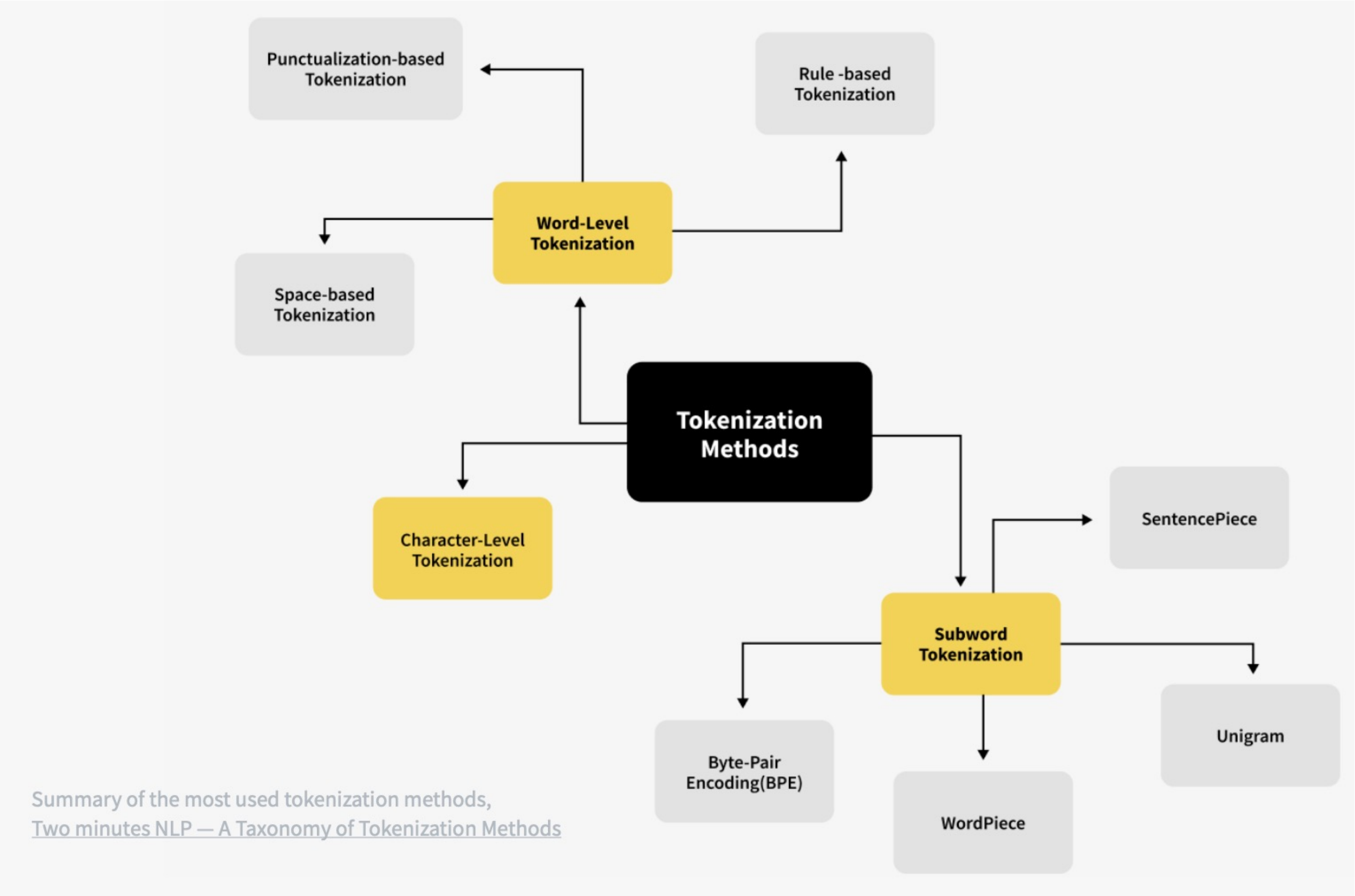
Why we need tokenization ?

- More general than words (e.g. to handle typos)
- Shorter sequences than with characters => allow smaller context window

Key Idea of Tokenizer:

- See Tokens as Common Subsequences

Common Methods of Tokenization



Tokenization Methods

Tokenization Methods	**Word-based tokenization**	**Character-based tokenization**	**Subword-based tokenization**
Example Tokenizers	Space tokenization (split sentences by space); rule-based tokenization (e.g. Moses, spaCy)	Character tokenization (simply tokenize on every character)	Byte-Pair Encoding (BPE); WordPiece; SentencePiece; Unigram (tokenizing by parts of a word vs. the entirety of a word; see table above)
Considerations	• **Downside:** Generates a very large vocabulary leading to a huge embedding matrix as the input and output layer; large number of out-of-vocabulary (OOV) tokens; and different meanings of very similar words • Transformer models normally have a vocabulary of less than 50,000 words, especially if they are trained only on a single language	• Lead to much smaller vocabulary; no OOV (out of vocabulary) tokens since every word can be assembled from individual characters • **Downside:** Generates very long sequences and less meaningful individual tokens, making it harder for the model to learn meaningful input representations. However, if character-based tokenization is used on non-English language, a single character could be quite information rich (like "mountain" in Mandarin).	• Subword-based tokenization methods follow the principle that frequently used words should not be split into smaller subwords, but rare words should be decomposed into meaningful subwords • **Benefit:** Solves the downsides faced by word-based tokenization and character-based tokenization and achieves both reasonable vocabulary size with meaningful learned context-independent representations.

Word-level Tokenization

Method:

- Rule-based – split text by spaces, punctuation and other hand-written rules

Challenges:

- Open Vocabulary Problem
  - Many words may never appear in training data. They become [UNK]
  - This is more severe in some languages, e.g. languages that concatenate words
- Typo'ed words also get tokenized to [UNK]

Character-level Tokenization

- When treating characters as your basic units, unknown (sub)tokens can still exist

Example:

If your basic units are [A-Za-z], Chinese characters cannot be tokenized.

Solution:

Byte-level encoding, e.g. BPE, that uses raw bytes (e.g. Unicode bytes, as basic character set

Subword modeling

Sample Data:

"This is tokenizing."

Character Level

[T] [h] [i] [s] [i] [s] [t] [o] [k] [e] [n] [i] [z] [i] [n] [g] [.]

Word Level

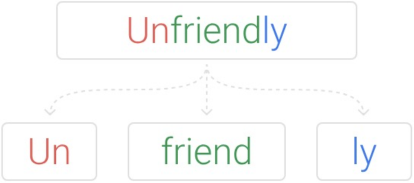
[This] [is] [tokenizing] [.]

Subword Level

[This] [is] [token] [izing] [.]

Subword modeling

- Subword modeling in NLP encompasses a wide range of methods for reasoning about structure below the word level. (Parts of words, characters, bytes.)



- The dominant modern paradigm is to learn a vocabulary of parts of words (subword tokens).
- At training and testing time, each word is split into a sequence of known subwords.

Advantages:

- Vocabulary is built dynamically, with controlled vocabulary size – a pre-defined hyperparameter as a design choice
  - Frequent words key whole and get assigned their own token
  - Rare words are split into sub-words ; more observations on sub-words
  - Utilization of morphology information

Subword modeling-based Tokenization methods

- Byte-Pair Encoding (BPE) [Gage 1994]:
  - originally used for Machine Translation
- WordPiece
- Unigram
- SentencePiece

Subword-based Tokenization Methods	Byte-Pair Encoding (BPE)	WordPiece	Unigram	SentencePiece
Description	One of the most popular subword tokenization algorithms. The Byte-Pair-Encoding works by starting with characters, while merging those that are the most frequently seen together, thus creating new tokens. It then works iteratively to build new tokens out of the most frequent pairs it sees in a corpus. BPE is able to build words it has never seen by using multiple subword tokens, and thus requires smaller vocabularies, with less chances of having "unk" (unknown) tokens.	Very similar to BPE. The difference is that WordPiece does not choose the highest frequency symbol pair, but the one that maximizes the likelihood of the training data once added to the vocabulary (evaluates what it loses by merging two symbols to ensure it's worth it)	In contrast to BPE / WordPiece, Unigram initializes its base vocabulary to a large number of symbols and progressively trims down each symbol to obtain a smaller vocabulary. It is often used together with SentencePiece.	The left 3 tokenizers assume input text uses spaces to separate words, and therefore are not usually applicable to languages that don't use spaces to separate words (e.g. Chinese). SentencePiece treats the input as a raw input stream, thus including the space in the set of characters to use. It then uses the BPE / Unigram algorithm to construct the appropriate vocabulary.
Considerations	BPE is particularly useful for handling rare and out-of-vocabulary words since it can generate subwords for new words based on the most common character sequences. Downside: BPE can result in subwords that do not correspond to linguistically meaningful units.	WordPiece can be particularly useful for languages where the meaning of a word can depend on the context in which it appears.	Unigram tokenization is particularly useful for languages with complex morphology and can generate subwords that correspond to linguistically meaningful units. However, unigram tokenization can struggle with rare and out-of-vocabulary words.	SentencePiece can be particularly useful for languages where the meaning of a word can depend on the context in which it appears.

Byte Pair Encoding (BPE) and Unigram Subword Tokenizers

Algorithm 1 Byte-pair encoding (Sennrich et al., 2016; Gage, 1994)

Input: set of strings D, target vocab size k
procedure BPE(D, k)
    $V \leftarrow$ all unique characters in D
        (about 4,000 in English Wikipedia)
    while $|V| < k$ **do** ▷ Merge tokens
        $t_L, t_R \leftarrow$ Most frequent bigram in D
        $t_{\text{NEW}} \leftarrow t_L + t_R$ ▷ Make new token
        $V \leftarrow V + [t_{\text{NEW}}]$
        Replace each occurrence of t_L, t_R in
            D with t_{NEW}
    end while
    return V
end procedure

Algorithm 2 Unigram LM (Kudo, 2018)

Input: set of strings D, target vocab size k
procedure UNIGRAMLM(D, k)
    $V \leftarrow$ all substrings occurring more than
        once in D (not crossing words)
    while $|V| > k$ **do** ▷ Prune tokens
        Fit unigram LM θ to D
        for $t \in V$ **do** ▷ Estimate token 'loss'
            $L_t \leftarrow p_\theta(D) - p_{\theta'}(D)$
            where θ' is the LM without token t
        end for
        Remove $\min(|V| - k, \lfloor \alpha |V| \rfloor)$ of the
        tokens t with highest L_t from V,
        where $\alpha \in [0, 1]$ is a hyperparameter
    end while
    Fit final unigram LM θ to D
    return V, θ
end procedure

For details, see https://huggingface.co/learn/nlp-course/en/chapter6/7

BPE is 'bottom-up' (merge characters). Unigram is 'top-down' (prune substrings)

Example of Byte Pair Encoding (BPE)

Steps:

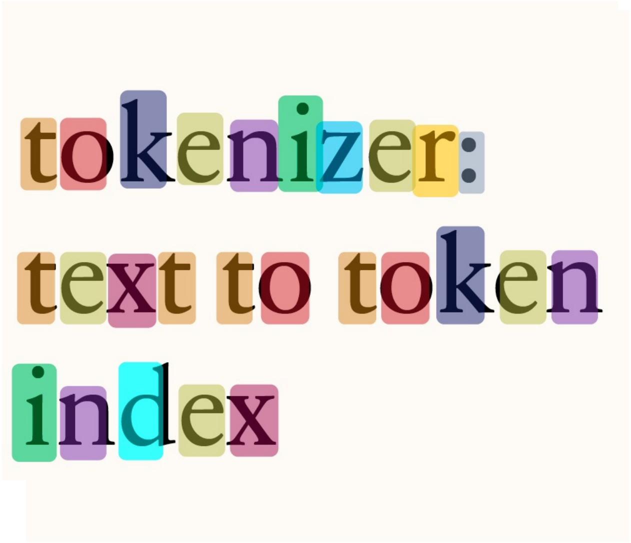
1. Take Large Corpus of Text.

tokenizer:
text to token
index

Example of Byte Pair Encoding (BPE)

Steps:

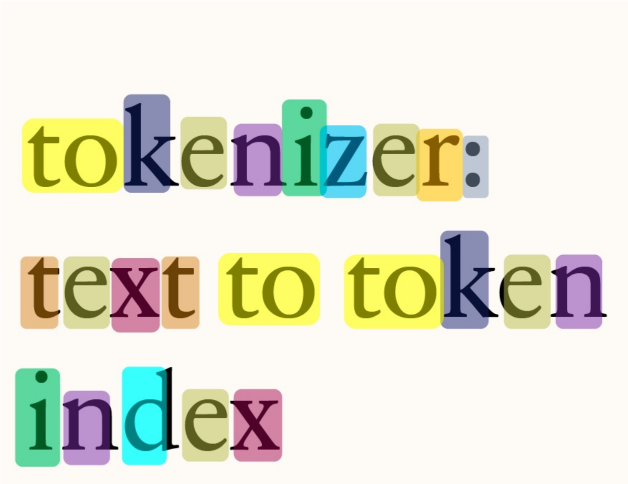
1. Take Large Corpus of Text.
2. Start with One Token per Character



Example of Byte Pair Encoding (BPE)

Steps:

1. Take Large Corpus of Text.
2. Start with One Token per Character
3. Merge Common Pairs of Tokens into a Token



Example of Byte Pair Encoding (BPE)

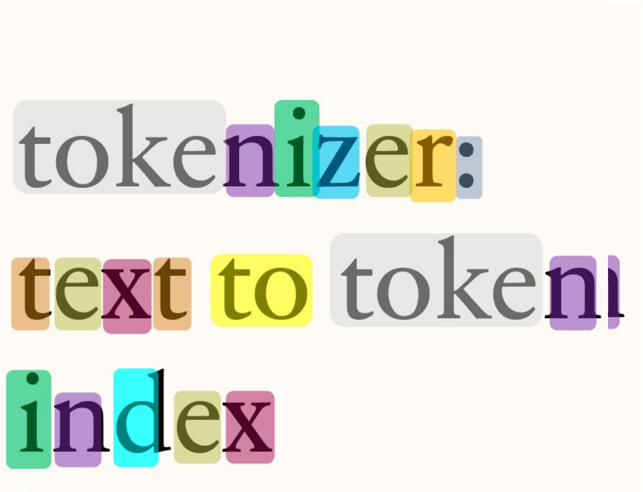
Steps:

1. Take Large Corpus of Text.
2. Start with One Token per Character
3. Merge Common Pairs of Tokens into a Token
4. Repeat until the desirable vocab size has been reached or all merged

Example of Byte Pair Encoding (BPE)

Steps:

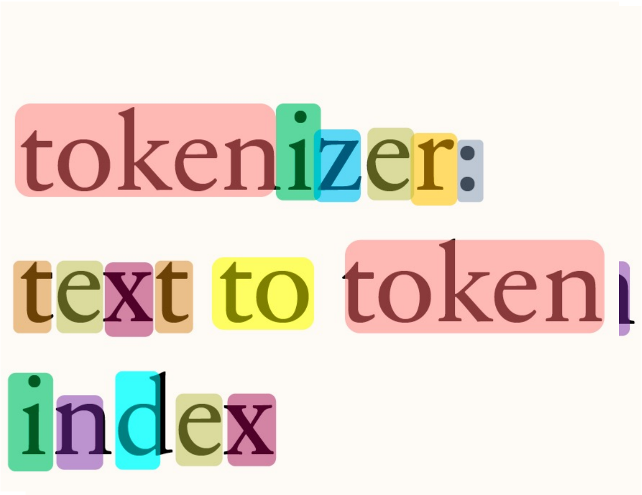
1. Take Large Corpus of Text.
2. Start with One Token per Character
3. Merge Common Pairs of Tokens into a Token
4. Repeat until the desirable vocab size has been reached or all merged



Example of Byte Pair Encoding (BPE)

Steps:

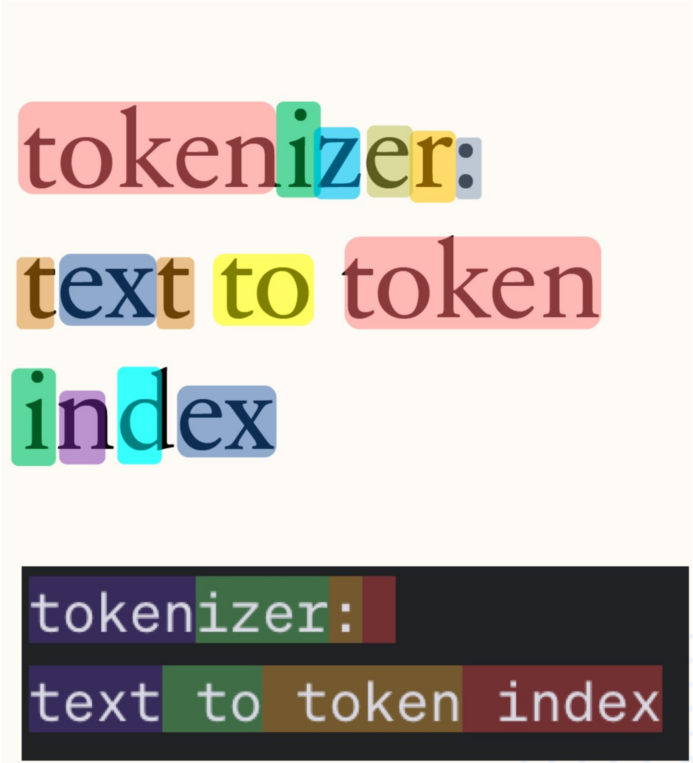
1. Take Large Corpus of Text.
2. Start with One Token per Character
3. Merge Common Pairs of Tokens into a Token
4. Repeat until the desirable vocab size has been reached or all merged



Example of Byte Pair Encoding (BPE)

Steps:

1. Take Large Corpus of Text.
2. Start with One Token per Character
3. Merge Common Pairs of Tokens into a Token
4. Repeat until the desirable vocab size has been reached or all merged



Unigram Tokenizers

Original:	furiously	**Original:**	tricycles	**Original:**	nanotechnology
BPE:	_fur iously	**BPE:**	_t ric y cles	**BPE:**	_n an ote chn ology
Uni. LM:	_fur ious ly	**Uni. LM:**	_tri cycle s	**Uni. LM:**	_nano technology

Original:	Completely preposterous suggestions
BPE:	_Comple t ely _prep ost erous _suggest ions
Unigram LM:	_Complete ly _pre post er ous _suggestion s

Original:	corrupted	**Original:**	1848 and 1852,
BPE:	_cor rupted	**BPE:**	_184 8 _and _185 2,
Unigram LM:	_corrupt ed	**Unigram LM:**	_1848 _and _1852 ,

Original	磁性は様々に分類がなされている。						
BPE	磁	性は		様々	に分類		がなされている	。
Unigram LM	磁	性	は	様々	に	分類	がなされている	。
Gloss	magnetism		(top.)	various ways	in	classification	is done	.
Translation	Magnetism is classified in various ways.							

Some (Bostrom and Durrett 2020) have argued that BPE produces less semantic tokens.

.. But BPE based LMs do work fine – the transformer on top can do quite a bit.

Example of a Bad Tokenizer LLaMA for Chinese

Table 1: Tokenizer comparisons between original LLaMA and Chinese LLaMA.

	Length	Content
Original Sentence	28	人工智能是计算机科学、心理学、哲学等学科融合的交叉学科。
Original Tokenizer	35	‘_’, ‘人’, ‘工’, ‘智’, ‘能’, ‘是’, ‘计’, ‘算’, ‘机’, ‘科’, ‘学’, ‘、’, ‘心’, ‘理’, ‘学’, ‘、’, ‘0xE5’, ‘0x93’, ‘0xB2’, ‘学’, ‘等’, ‘学’, ‘科’, ‘0xE8’, ‘0x9E’, ‘0x8D’, ‘合’, ‘的’, ‘交’, ‘0xE5’, ‘0x8F’, ‘0x89’, ‘学’, ‘科’, ‘。’
Chinese Tokenizer	16	‘_’, ‘人工智能’, ‘是’, ‘计算机’, ‘科学’, ‘、’, ‘心理学’, ‘、’, ‘哲学’, ‘等’,‘学科’, ‘融合’, ‘的’, ‘交叉’, ‘学科’, ‘。’

Source:
Yiming Cui. et.al. EFFICIENT AND EFFECTIVE TEXT ENCODING FOR CHINESE LLAMA AND ALPACA. https://arxiv.org/pdf/2304.08177.pdf

Tokenizers in Practice

The non-google world uses BPE. Google uses the SentencePiece library, which (sometimes) refers to a non-BPE subword tokenizer

Model	Tokenizer
Original transformer	BPE
GPT 1/2/3	BPE
T5 / mT5 / T5v1.1	SentencePiece (Unigram)
Gopher/Chinchilla	SentencePiece (??)
PaLM	SentencePiece (??)
LLaMA	BPE

Important property – all of these tokenizers are *invertible*

SentencePiece

Open-source library with many subword tokenizers

Feature	SentencePiece	subword-nmt	WordPiece
Supported algorithm	BPE, unigram, char, word	BPE	BPE*
OSS?	Yes	Yes	Google internal
Subword regularization	Yes	No	No
Python Library (pip)	Yes	No	N/A
C++ Library	Yes	No	N/A
Pre-segmentation required?	No	Yes	Yes
Customizable normalization (e.g., NFKC)	Yes	No	N/A
Direct id generation	Yes	No	N/A

We will talk a bit about **normalization** and **unigram** subword tokenization

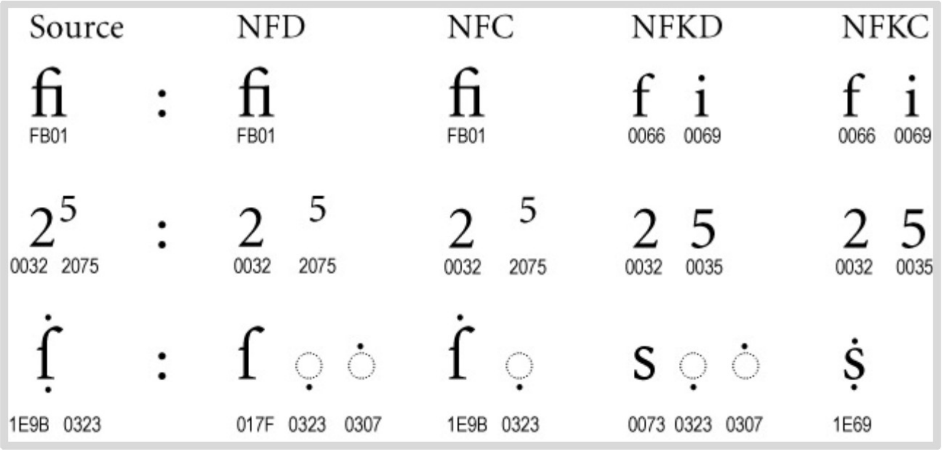
References: https://github.com/google/sentencepiece ; https://github.com/openai/tiktoken

NFKC Normalization

There are many characters that are different in Unicode but look very similar

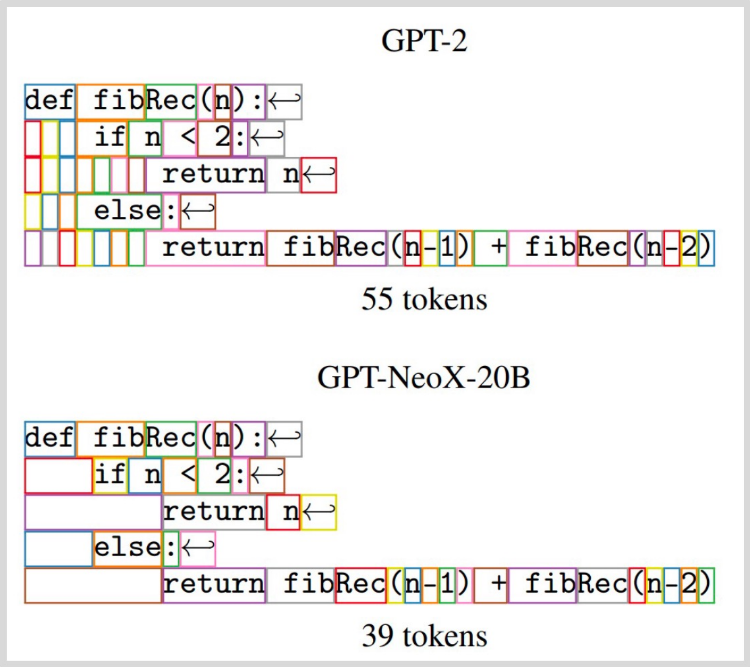
Roman ‘A’	Fullwidth ‘A’
A	Ａ

Some processing systems (e.g. sentencepiece) will NFKC normalize texts – with pros and cons



Whitespace and Number related Hacks

Multi-whitespace tokenization (GPT-NeoX)



Individual digit tokenization (LLaMA/DeepSeek)

Tokenizer. We tokenize the data with the byte-pair encoding (BPE) algorithm (Sennrich et al., 2015), using the implementation from Sentence-Piece (Kudo and Richardson, 2018). Notably, we split all numbers into individual digits, and fallback to bytes to decompose unknown UTF-8 characters.

Typical Vocabulary Sizes

Monolingual models – 30-50k vocab

Model	Token count
Original transformer	37000
GPT	40257
GPT2/3	50257
T5/T5v1.1	32128
LLaMA	32000

Multilingual / production systems 100-250k

Model	Token count
mT5	250000
PaLM	256000
GPT4	100276
BLOOM	250680
DeepSeek	100000
Qwen 15B	152064
Yi	64000

Monolingual vocabs don’t need to be huge, but multilingual ones do

Some Sample Dataset sizes in # of Tokens

- PALM: 780 billion tokens
- GLAM: 1.6 trillion
- Gopher: 300 billion
- GPT-3: 300 billion
- Chinchilla: 1.4 Trillion tokens

A. Training dataset

In Table A1 we show the training dataset makeup used for *Chinchilla* and all scaling runs. Note that both the *MassiveWeb* and Wikipedia subsets are both used for more than one epoch.

	Disk Size	Documents	Sampling proportion	Epochs in 1.4T tokens
MassiveWeb	1.9 TB	604M	45% (48%)	1.24
Books	2.1 TB	4M	30% (27%)	0.75
C4	0.75 TB	361M	10% (10%)	0.77
News	2.7 TB	1.1B	10% (10%)	0.21
GitHub	3.1 TB	142M	4% (3%)	0.13
Wikipedia	0.001 TB	6M	1% (2%)	3.40

Table A1 | ***MassiveText* data makeup.** For each subset of *MassiveText*, we list its total disk size, the number of documents and the sampling proportion used during training—we use a slightly different distribution than in Rae et al. (2021) (shown in parenthesis). In the rightmost column show the number of epochs that are used in 1.4 trillion tokens.

Chinchilla

Total dataset size = 780 billion tokens	
Data source	**Proportion of data**
Social media conversations (multilingual)	50%
Filtered webpages (multilingual)	27%
Books (English)	13%
GitHub (code)	5%
Wikipedia (multilingual)	4%
News (English)	1%

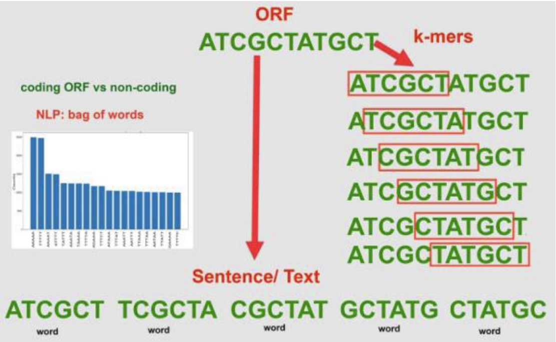
Palm

	Disk Size	Documents	Tokens	Sampling proportion
MassiveWeb	1.9 TB	604M	506B	48%
Books	2.1 TB	4M	560B	27%
C4	0.75 TB	361M	182B	10%
News	2.7 TB	1.1B	676B	10%
GitHub	3.1 TB	142M	422B	3%
Wikipedia	0.001 TB	6M	4B	2%

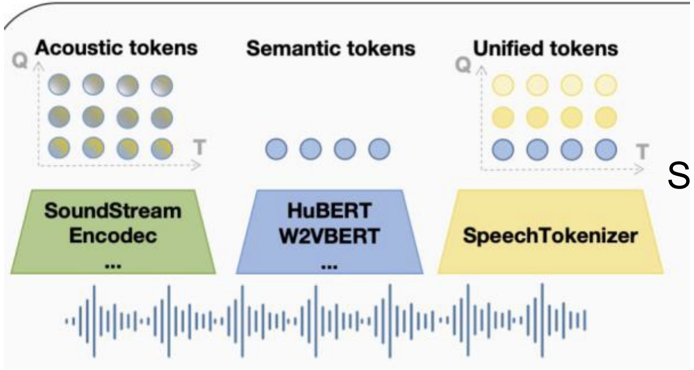
Table 2 | **MassiveText data makeup.** For each subset of *MassiveText*, we list its total disk size, its number of documents, and its number of SentencePiece tokens. During training we sample from *MassiveText* non-uniformly, using the sampling proportion shown in the right-most column.

Gopher

A Broader Sense of "Token"



Genes



Speech tokens

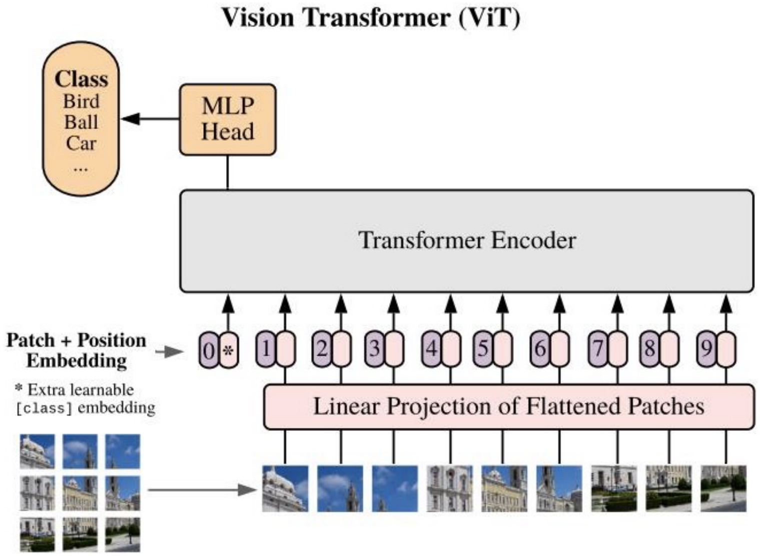


Image tokens

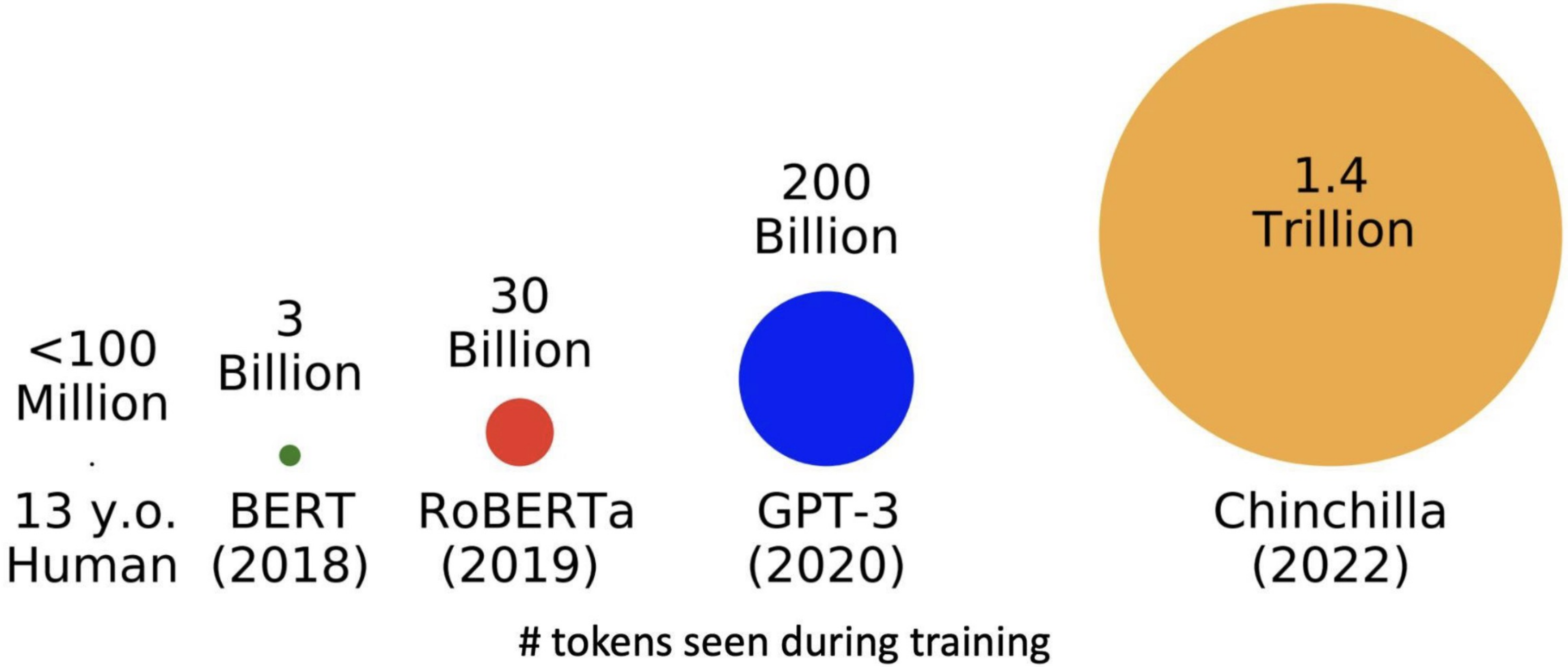
Source: Alexey Dosovitskiy. et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. https://arxiv.org/abs/2010.11929
Xin zhang et.al. SpeechTokenizer: Unified Speech Tokenizer for Speech Language Models. https://0nutation.github.io/SpeechTokenizer.github.io/

Tokenizer Summary

- Everyone uses invertible subword tokenizers (BPE, Unigram) for good reason
- NFKC normalization is a double edged sword (2^5) and many models don’t use it
- For math and code, careful manual handling of whitespace and numbers can help

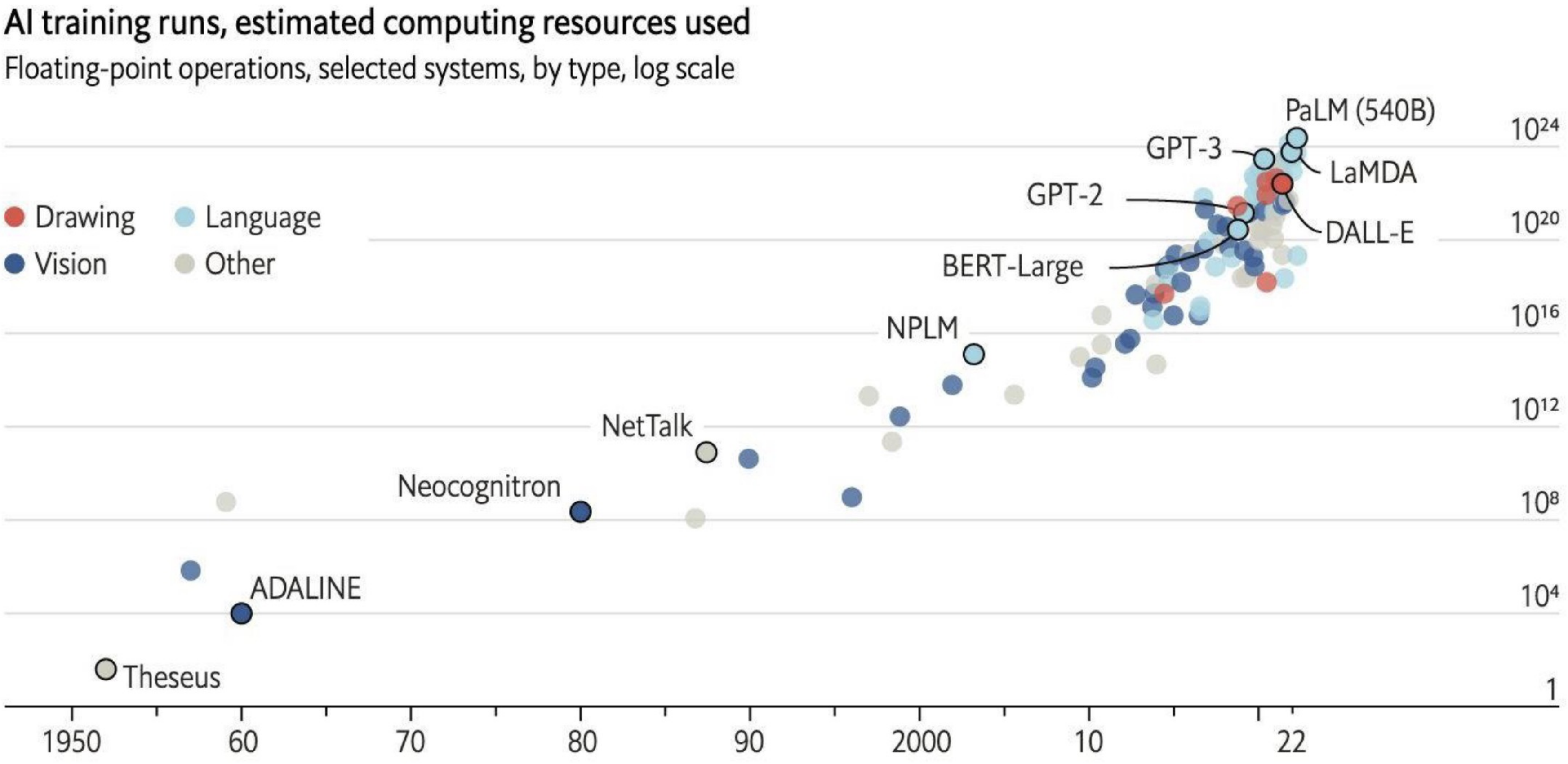
How Large is Large: No. of Tokens (D) for training LLMs

Large Language Models - Trillions of Tokens



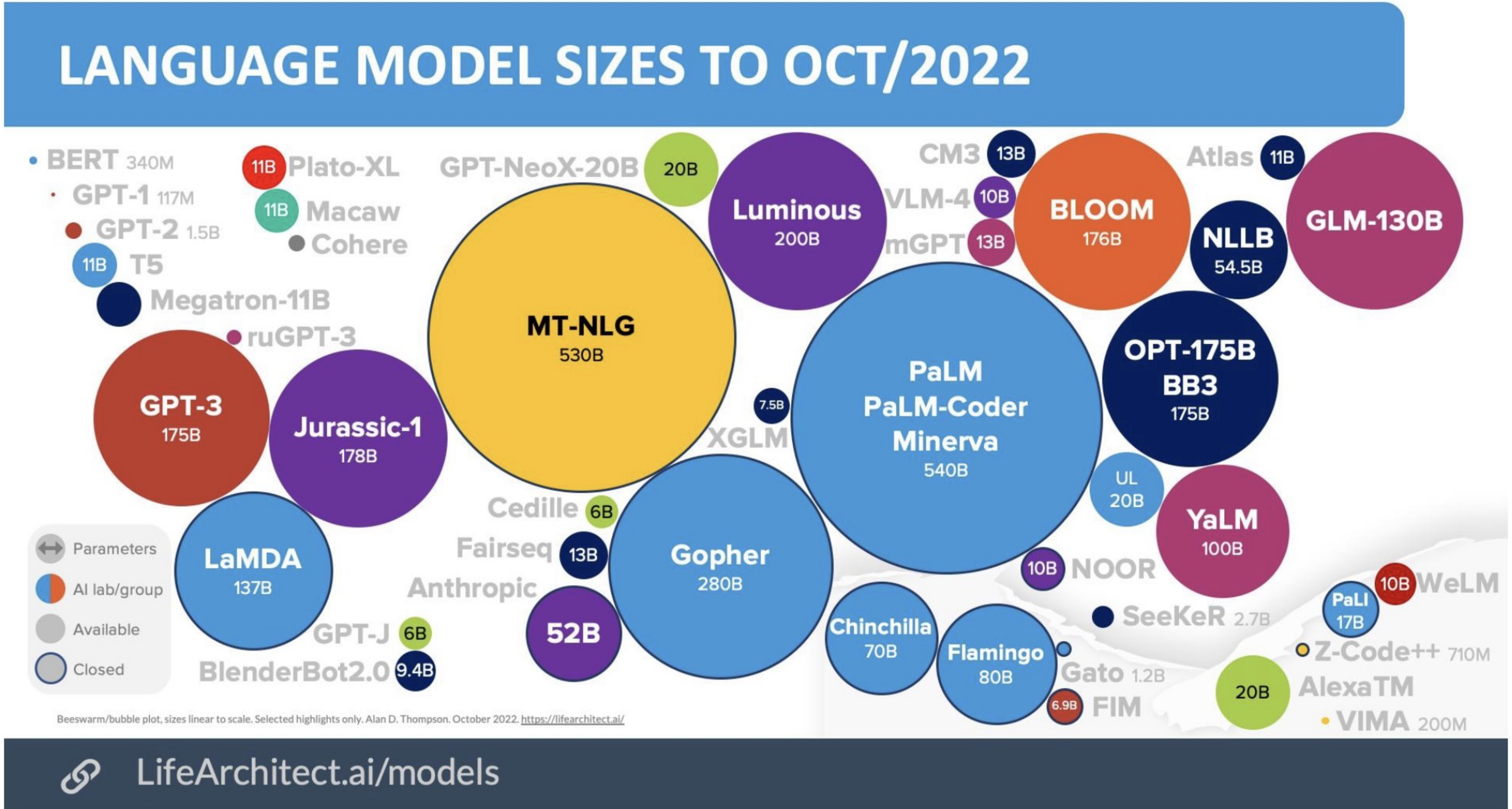
https://babylm.github.io/

How Large is Large ?



https://web.stanford.edu/class/cs224n/slides/cs224n-2023-lecture11-prompting-rlhf.pdf 1 yotta = 10^{24} **FLOPs: floating point operations**

How Large is Large: No. of Parameters (N) in LLMs

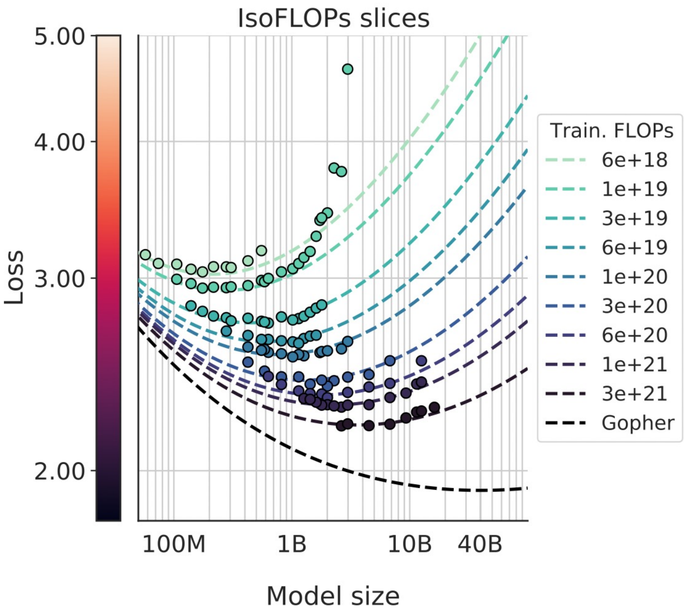


LLM Scaling Laws

Performance of LLMs is a smooth, well-behaved, predictable function of:
- **N**, the number of parameters in the network
- **D**, the amount of text we train on

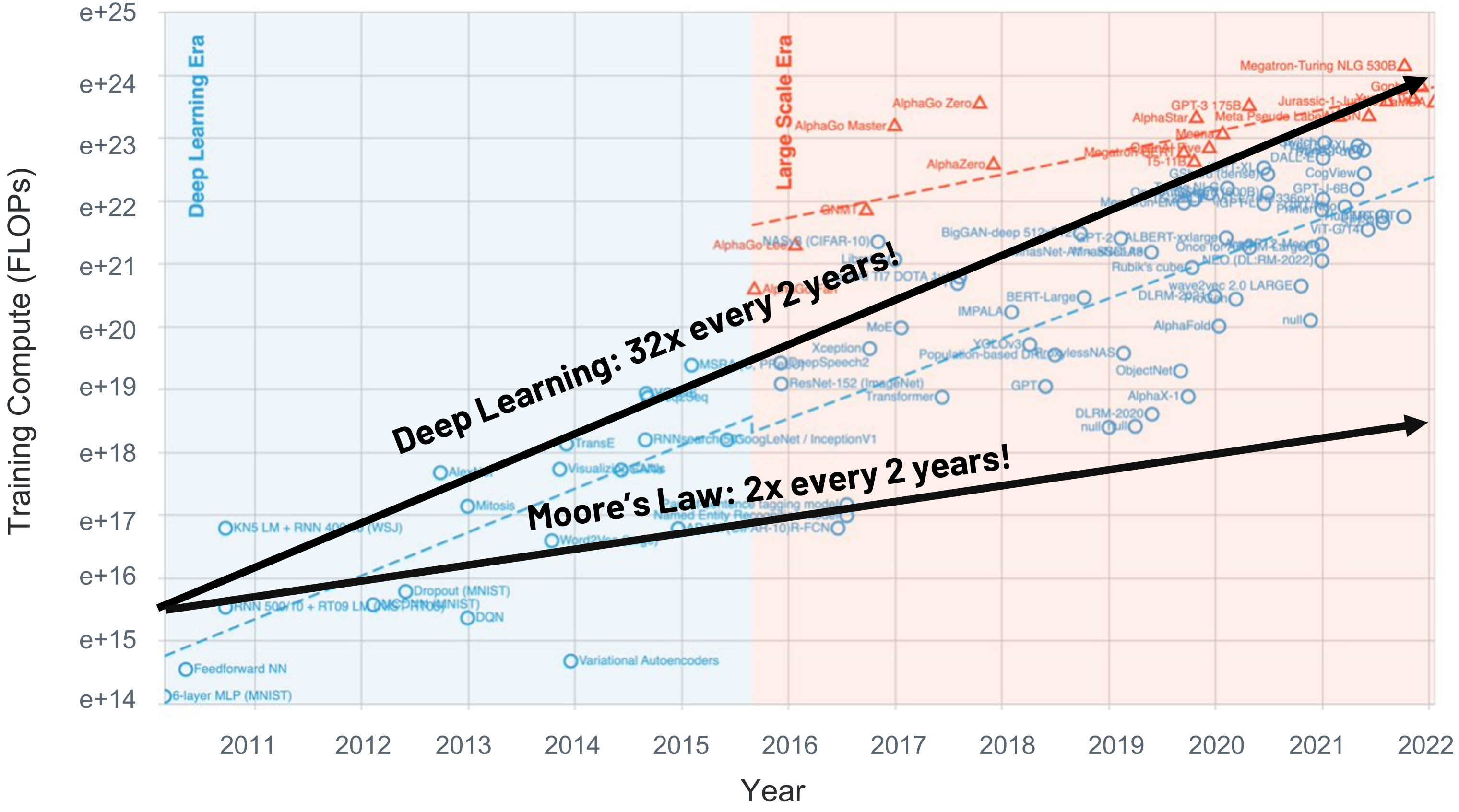
And the trends do not show signs of “topping out”

=> We can expect more intelligence by scaling



[Training Compute-Optimal Large Language Models]

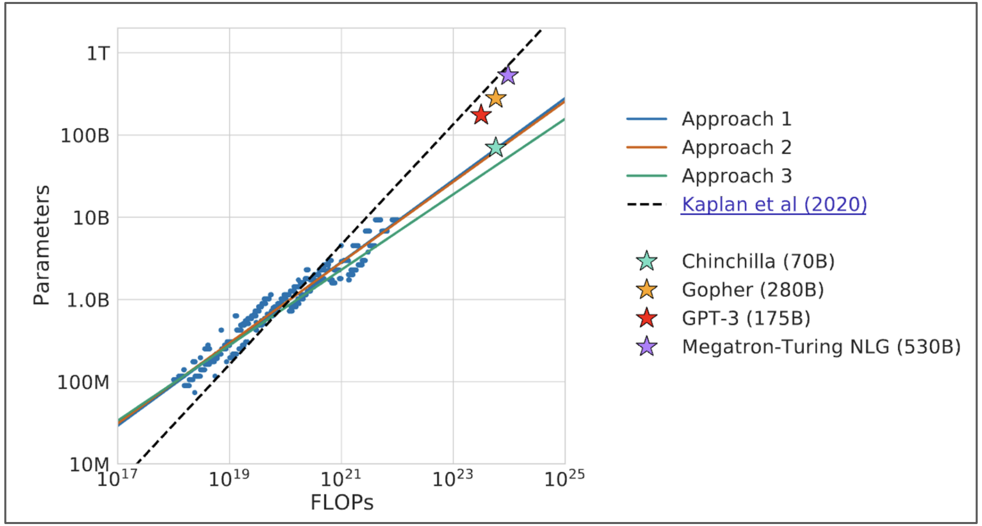
Large Computation Cost: AI models vs. Moore's Law



Sevilla et al., "Compute Trends Across Three Eras of Machine Learning", 2022

Pre-Training: Scaling Laws

Given a fixed compute budget, what is the optimal model size and training dataset size for training a transformer LM?

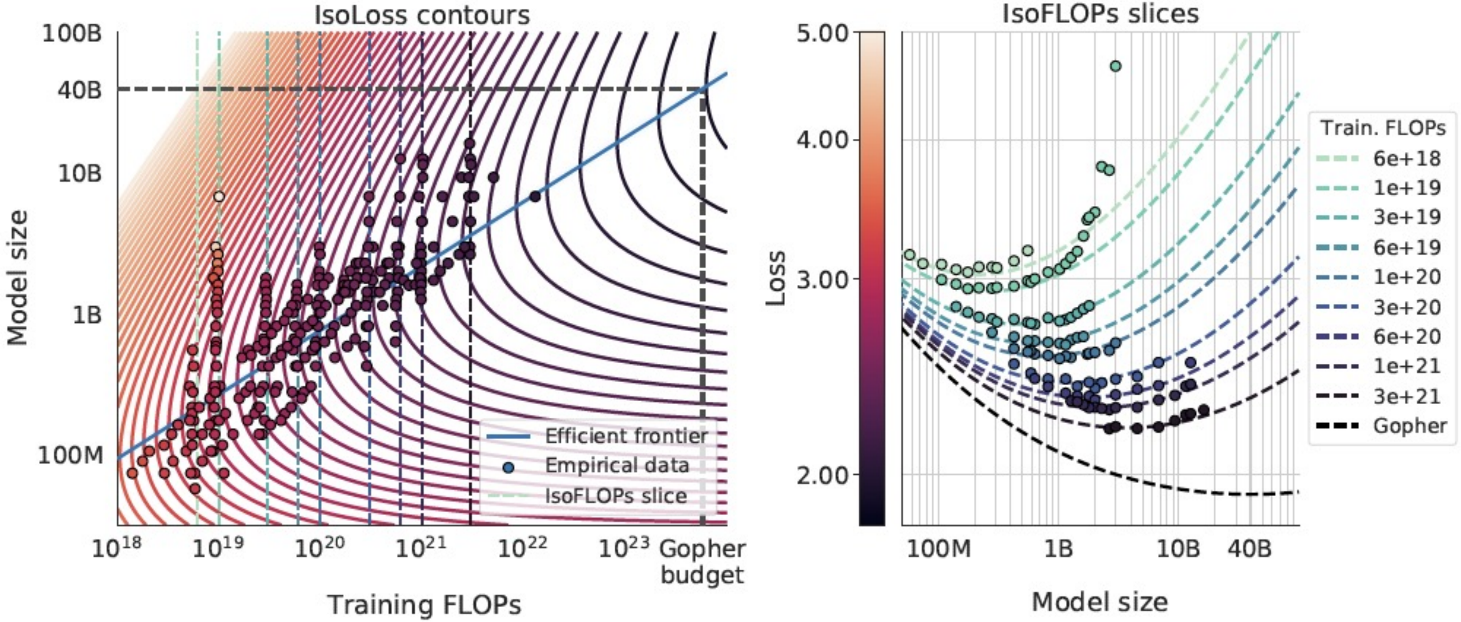


Chinchilla Scaling Law:

For every doubling of model size, the number of training tokens must also be doubled.

J. Hoffmann et al. [Training Compute-Optimal Large Language Models](). 2022.

LLM Scaling Laws



Source: Training Compute-Optimal Large Language Models, DeepMind, https://arxiv.org/pdf/2203.15556, Mar 2022.

Figure 4 | **Parametric fit.** We fit a parametric modelling of the loss $\hat{L}(N, D)$ and display contour (**left**) and isoFLOP slices (**right**). For each isoFLOP slice, we include a corresponding dashed line in the left plot. In the left plot, we show the efficient frontier in blue, which is a line in log-log space. Specifically, the curve goes through each iso-loss contour at the point with the fewest FLOPs. We project the optimal model size given the *Gopher* FLOP budget to be 40B parameters.

For constant amount of Training Compute, **Optimal Ratio** of **D/ N ~= 20**

LLM Scaling Laws

- Before 2022, most of the largest LLMs were "Under Trained" (with D/N << 20)

Model	Size (# Parameters)	Training Tokens
LaMDA (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinchilla	70 Billion	1.4 Trillion

Source: Training Compute-Optimal Large Language Models, DeepMind, https://arxiv.org/pdf/2203.15556, Mar 2022.

BILLBOARD CHART FOR LANGUAGE MODELS JUN/ 2024

MODELS

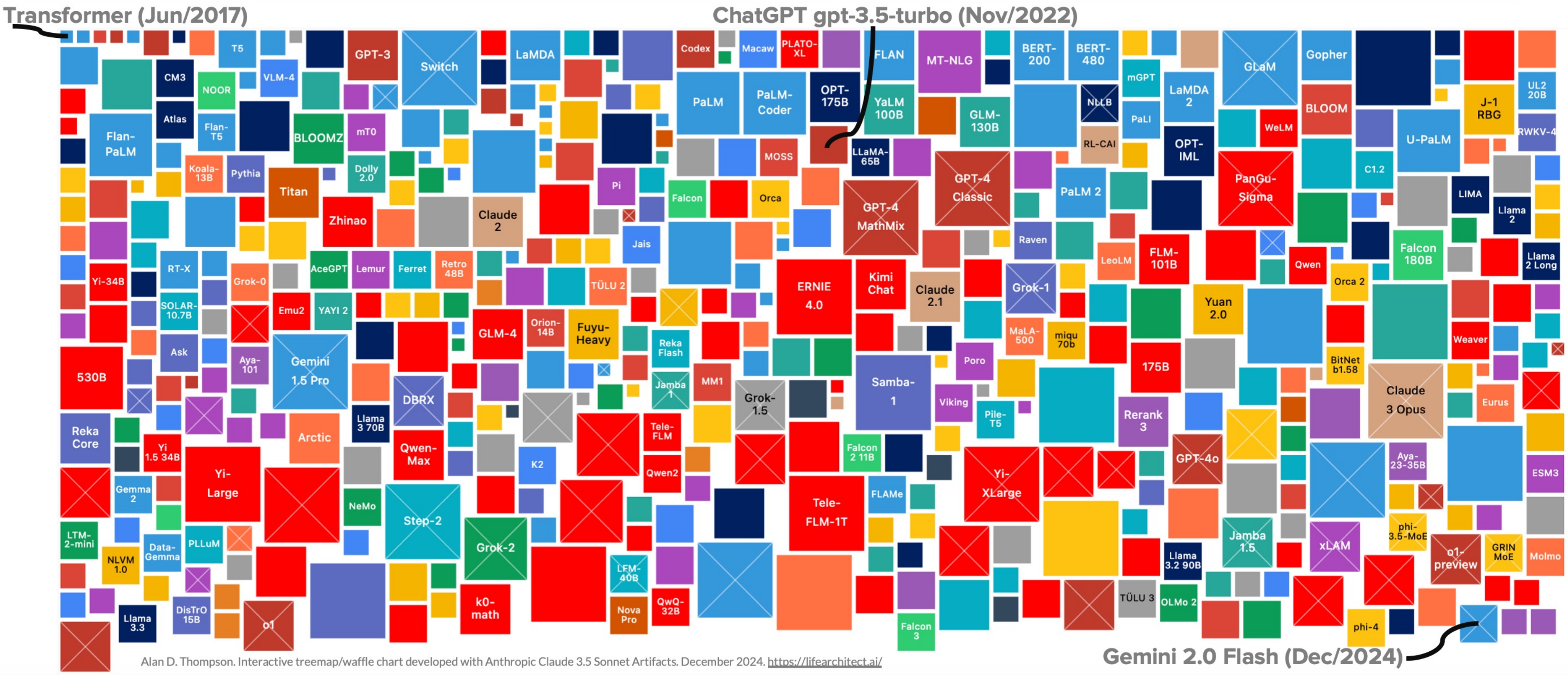
Now (Jun/2024)	6m ago (Dec/2023)	12m ago (Jun/2023)	ALScore	Model name Details	AI lab Openness
1	–	–	29.8	**Claude 3 Opus** 2T trained on 40T tokens*	◆ Anthropic API
2	1	–	22.4	**Gemini Ultra 1.0** 1.5T trained on 30T tokens*	◆ Google DM API
3	–	–	22.4	**Gemini 1.5 Pro** 1.5T trained on 30T tokens*	◆ Google DM API
4	–	–	21.1	**Yi-XLarge** 2T trained on 20T tokens*	◆ 01-ai API
5	–	–	16.3	**Inflection-2.5** 1.2T on 20T tokens*	◆ Inflection AI API
6	2	1	15.9	**GPT-4 (family)** 1.7T trained on 13T tokens*	◆ OpenAI API
7	3	–	14.9	**ERNIE 4.0** 1T trained on 20T tokens*	◆ Baidu API
8	–	–	8.2	**SenseNova 5.0** 600B on 10T tokens	◆ SenseTime API

DATASETS

Now (Jun/2024)	6m ago (Dec/2023)	12m ago (Jun/2023)	Size (TB)	Dataset name Details	AI lab Language
1	1	–	130	**Gemini** 30T tokens in 130TB*	◆ Google DM Multilingual
2	2	–	125	**RedPajama-Data-v2** 30T tokens in 125TB	◆ Together AI Multilingual
3	3	1	86	**Piper monorepo** 37.9T tokens in 86TB	◆ Google Code
4	4	–	40	**Massive Never-ending BT Vast Chinese corpus** 30T/40TB	◆ MNBVC Chinese
5	–	–	44	**FineWeb** 15T tokens in 44TB	◆ HF English
6	5	2	40	**GPT-4** 13T tokens in 40TB*	◆ OpenAI English
7	–	–	31.5	**FineWeb-Edu-score-2** 5.4T tokens in 31.5TB	◆ HF English
8	6	–	27	**CulturaX** 6.3T tokens in 27TB	◆ UOregon Multilingual

Selected highlights only, some older models disregarded. * = estimates and hypothesis only based on current information. Alan D. Thompson. June 2024. https://lifearchitect.ai/

LARGE LANGUAGE MODEL HIGHLIGHTS 2017–2024



Alan D. Thompson. Interactive treemap/waffle chart developed with Anthropic Claude 3.5 Sonnet Artifacts. December 2024. https://lifearchitect.ai/

LLMs Data from LifeArchitect.ai (Jan 2025)

Model	Lab	Parameters (B)	Tokens trained (B)	Ratio Tokens:Params(Chinchilla scaling≥20:1)	ALScore "ALScore" is a quick and dirty	MMLU			Training dataset	Announced▼	Public?	Arch	Notes
AuroraGPT (ScienceGPT	Argonne National La	2000	30000	15						TBA			Three models targeted in Jul/2024: AuroraGPT-7B-P (Ponte Vecchio GPU testing) AuroraGPT-7B-A (Aurora) AuroraG
Grok-3	xAI	27000								TBA		MoE	Training Jul-Dec 2024 on 100,000 H100s. Due December 2024.
GPT-5	OpenAI	27000	114000							TBA		MoE	Due 2025.
GPT-6		https://lifearchitect.ai/gpt-6/								TBA			Due 2025.
Llama 4	Meta AI	https://x.com/Ahmad_Al_Dahle/status/1851822285377933809								TBA		Dense	Training Oct/2024-Feb/2025 on 100,000 H100s. Due 2025.
~~MAI-1~~	~~Microsoft~~	~~500~~	~~10000~~	~~20~~	~~7.5~~					~~TBA~~		~~Dense~~	Potential failed training run 2024. MAI=Microsoft artificial intelligence. MSFT CTO statement: https://archive.md/XR
o4	OpenAI	https://lifearchitect.ai/o4/								TBA			Due 2025.
Sonus-1 Reasoning	Rubik's AI	405	15000	38	4.6	90.15	73.1	67		Jan/2025		Dense	Likely a Llama 3.1 405B wrapper. ALPrompt 2024H1=5/5. ALPrompt 2024H2=2/5. ALPrompt 2025H1=1/5. This is a st
YuLan-Mini	Renmin	2.4	1080	450	0.2	51.79				Dec/2024		Dense	"1.08T tokens for training. Among them are 481B English web data, 138B general English knowledge, 227B code pre
DeepSeek-V3	DeepSeek-AI	685	14800	22	4.6	87.1	64.4	59		Dec/2024		MoE	37B active. Announce: https://github.com/deepseek-ai/DeepSeek-V3?tab=readme-ov-file
EON-8B	LinkedIn	8	15000	1875	1.2					Dec/2024		Dense	"We found the EON-8B model (a domain-adapted Llama 3.1-8B variant) to be 75x and 6x cost effective in comparis
o3	OpenAI	5000	100000	20	74.5			88		Dec/2024		MoE	SoTA model for Dec/2024. Parameter estimate is very rough centrepoint for range 400B-52T.
RWKV-7 Goose	RWKV	0.4	332	830	0.0					Dec/2024		Dense	RWKV (pronounced RwaKuv) is an RNN: "multilingual, supporting over 100 languages and code.". Full run is 332B to
ModernBERT	International	0.395	2000	5064	0.1					Dec/2024		Dense	"a proper workhorse model, for retrieval, classification, etc." https://bsky.app/profile/howard.fm/post/3ldod2afps6
Granite 3.1 8B	IBM	8	12000	1500	1.0					Dec/2024		Dense	
Bamba-9B	CMU	9	2200	245	0.5	60.77	17.53	4.1		Dec/2024		Dense	"trained by IBM, Princeton, CMU, and UIUC on completely open data. At inference time, the model demonstrates 2.
o1-2024-12-17	OpenAI	200	20000	100	0.7	91.8		76		Dec/2024		MoE	"o1-2024-12-17 sets new state-of-the-art results on several benchmarks, improving cost-efficiency and performance
Falcon 3	TII	10	16000	1600	0.7	73.1	42.5	34		Dec/2024		Dense	"We conducted a single large-scale pretraining run on the 7B model, using 1024 H100 GPU chips, leveraging 14 trilli
Command R7B	Cohere	7	2000	286	0.4		28.5	7.7		Dec/2024		Dense	
Maya	Cohere	8	4800	600	0.7					Dec/2024		Dense	VLM.
BLT	Meta AI	8	4500	563	0.6	57.4				Dec/2024		Dense	Byte Latent Transformer (BLT), a new byte-level LLM architecture that, for the first time, matches tokenization-base
Large Concept Model	Meta AI	7	2700	563	0.6					Dec/2024		Dense	"autoregressive sentence prediction in an embedding space." 7.7T tokens is a misprint, should be 2.2T as in paper.
phi-4	Microsoft	14	10000	715	1.2	84.8	70.4	56		Dec/2024		Dense	Available on HF from 22/Dec/2024.
Gemini 2.0 Flash exp	Google DeepMind	30	30000	1000	3.2	87	76.4	62		Dec/2024		MoE	Gemini 2.0 Flash was first model released, 11/Dec/2024. "New Modalities: Gemini 2.0 introduces native image ger
Moxin-7B	International	7	2000	286	0.4	60.97				Dec/2024		Dense	"Fully Open Source" with pre-training code, configurations, training and fine-tuning datasets, and intermediate chec
1T	Cerebras	1000	20000	20	14.9					Dec/2024		Dense	"For Sandia's trillion parameter training run, Cerebras configured a 55 terabyte MemoryX device."
InternVL 2.5	Shanghai AI Laborat	78	18120	233	4.0	86.1	71.1	49		Dec/2024		Dense	o1 reasoning model copy. Benchmarks are estimates based on Qwen2.5 72B Instruct as the base LLM (InternVL 2.5=
Llama 3.3	Meta AI	70	15000	215	3.4	86	68.9	51		Dec/2024		Dense	Drop-in replacement for Llama 3.1 70B, comparable performance to Llama 3.1 405B.
EXAONE-3.5	LG	32	6500	204	1.5	78.3		40		Dec/2024		Dense	"EXAONE"="EXpert AI for EveryONE". Training tokens/ratio dropped from EXAONE-3 7.8B with 8T (Aug/2024) to thi
Deepthought-8B	Ruliad	8	15000	1875	1.2					Dec/2024		Dense	o1 reasoning model copy. No evals. Llama 3.1 8B base.
Sailor2	Sail	20	18510	926	2.0					Dec/2024		Dense	SEA languages. Continual pretraining based on Qwen2.5. Project page: https://sea-sailor.github.io/blog/sailor2/
Pleias 1.0	PleIAs	3	1086	362	0.2					Dec/2024		Dense	Trained on the Jean Zay supercomputer, 192x H100s for 20 days. Dataset is new CC + Synthetic: https://huggingface
o1	OpenAI	200	20000	100	6.7	92.3	91	79		Dec/2024		MoE	"a version of our most intelligent model that thinks longer for the most reliable responses" System card about safet
Nova Pro	Amazon	90	10000	112	3.2	85.9		47		Dec/2024		Dense	Multimodal, same performance as Llama 3.2 90B ∴ est 90B. Model card was hidden: https://assets.amazon.scienc
DisTrO 15B	Nous Research	15	100	7	0.1	23.48				Dec/2024		Dense	"About 14 DGXes scattered around the globe. Sometimes more sometimes less, it varies depending on availability.
INTELLECT-1	Prime Intellect	10	1000	100	0.3	49.89		28		Nov/2024		Dense	Training complete 22/Nov/2024. Fully distributed training: "the first decentralized training run of a 10-billion-param
QwQ-32B	Alibaba	32	18000	563	2.5			65		Nov/2024		Dense	o1 reasoning model copy. Scores 1/5 on latest ALPrompt 2024 H2. Qwen with Question=QwQ
Teuken-7B	OpenGPT-X	7	4000	572	0.6	50				Nov/2024		Dense	24 EU languages (60% non-English): bg, cs, da, de, el, en, es, et, fi, fr, ga, hr, hu, it, lt, lv, mt, nl, pl, pt, ro, sk, sl, sv. h
OLMo 2	Allen AI	13	5600	431	0.9	68.6				Nov/2024		Dense	Open Language Model (OLMo) 2 Apache 2.0 license for research and educational use. Paper coming. Data: 5 trillion
Bi-Mamba	CMU	2.7	1260	467	0.2					Nov/2024		Dense	Unreleased, but will be replicated. "a scalable and powerful 1-bit Mamba architecture designed for more efficient l
k0-math	Moonshot AI	100	2000	20	1.5					Nov/2024		Dense	o1 reasoning model copy, maths only. Very little info available. Chinese. Long context. No paper.
Marco-o1	Alibaba	7	7000	1000	0.7					Nov/2024		Dense	o1 reasoning model copy. No evals. Qwen2-7B-Instruct with a combination of the filtered Open-O1 CoT dataset, Ma
TÜLU 3	Allen AI	70	15600	223	3.5	83.1	65.8	45		Nov/2024		Dense	Llama 3.1 post-training, worse performance on most benchmarks. Post training methods include new Reinforcemer
gpt-4o-2024-11-20		200	20000	100	6.7	85.7		46		Nov/2024		MoE	Material decrease in benchmark scores (GPQA: -13.37%, MMLU: -3.38%) compared to Aug/2024. Pruned? Quantize
DeepSeek-R1-Lite	DeepSeek-AI	67	2000	30	4.6			59		Nov/2024		Dense	o1 reasoning model copy. Scores 0/5 on latest ALPrompt 2024 H2 "DeepSeek-R1-Lite is currently still in the iterative
Xmodel-LM	XiaoduoAI	1.1	2064	1877	0.2	25.9				Nov/2024		Dense	SLM

Source: https://docs.google.com/spreadsheets/d/1kc262HZSMAWI6FVsh0zJwbB-ooYvzhCHaHcNUiA0_hY

Leaderboard of LLM Chatbot Arena

Crowdsourced platform where humans vote on pairwise comparisons of different LLMs (akin to Elo rating system in Chess).

Model	⭐ Arena Elo rating	📈 MT-bench (score)	MMLU	License
GPT-4-Turbo	1210	9.32		Proprietary
GPT-4	1159	8.99	86.4	Proprietary
Claude-1	1146	7.9	77	Proprietary
Claude-2	1125	8.06	78.5	Proprietary
Claude-instant-1	1106	7.85	73.4	Proprietary
GPT-3.5-turbo	1103	7.94	70	Proprietary
WizardLM-70b-v1.0	1093	7.71	63.7	Llama 2 Community
Vicuna-33B	1090	7.12	59.2	Non-commercial
OpenChat-3.5	1070	7.81	64.3	Apache-2.0
Llama-2-70b-chat	1065	6.86	63	Llama 2 Community
WizardLM-13b-v1.2	1047	7.2	52.7	Llama 2 Community
zephyr-7b-beta	1042	7.34	61.4	MIT
MPT-30B-chat	1031	6.39	50.4	CC-BY-NC-SA-4.0

What's Next: Post-Training

Pre-training vs. Finetuning/ Post-Training/ Adaptation

“Pre-trained” LLMs are trained solely based on next word prediction on vast amounts of text data (e.g. the internet).

They need further “finetuning” to be able to follow instructions and be useful!

Prompt	Response from a pre-trained model	Response from a finetuned model
Translate cheese from English to French	Translate cheese from English to Spanish Translate cheese from French to English	The French word for cheese is " fromage". The pronunciation is as follows: froh-MAHZH