Introduction to Large Language Models

Spring 2026

LLM Training:
Pre-Training

(Some slides adapted from Ralph Grishman at NYU,
Yejin Choi at UWashington, N. Tomura at UDepaul, Jurafsky and Martin, CS224N,
CS224, CME295 at Stanford and other resourses on the web)

The Three Core Steps of LLM Training

1. Pre-training (Foundation Stage)

The model learns general language patterns by predicting the next token on massive unlabeled datasets.

What happens:

- Train on trillions of tokens (web text, books, code, research papers).
- Objective: next-token prediction or masked-token prediction.
- Learns grammar, facts, reasoning patterns, semantics, world knowledge.
- Produces the **base model**.

Output:

A raw but powerful model that knows a lot, but does **not** follow instructions well.

The Three Core Steps of LLM Training

2. Supervised Fine-Tuning (SFT)

The model learns to follow instructions by training on human-written demonstration data.

What happens:

- Curated dataset of instruction–response pairs.
- Helps the model behave like a helpful assistant.
- Teaches format, reasoning style, tone, and task-following behavior.

Output:

A model that can follow prompts more reliably, but still may be inconsistent, unsafe, or hallucinate.

The Three Core Steps of LLM Training

3. Preference Alignment (RLHF or DPO)

The model learns to produce outputs humans *prefer*, using comparisons rather than instructions.

Two major methods:

✅ **RLHF (Reinforcement Learning from Human Feedback)**

- Humans rank multiple model responses.
- A reward model learns these preferences.
- The LLM is optimized to maximize reward (often via PPO).

✅ **DPO (Direct Preference Optimization)**

- No RL loop.
- Train the model directly on preferred vs dispreferred outputs.

Output:

A model that is:

- safer
- more polite
- more aligned with human preferences
- more stable in reasoning

Optional Steps

4. Post-training Optimization (Test-time compute, safety passes, etc.)
Not part of weight training — but done **during inference**:

- Chain of Thought prompting
- Self-consistency sampling
- Tool use / retrieval (RAG)
- Verification passes
- Guardrails and safety filters

These improve output quality **without modifying model weights**.

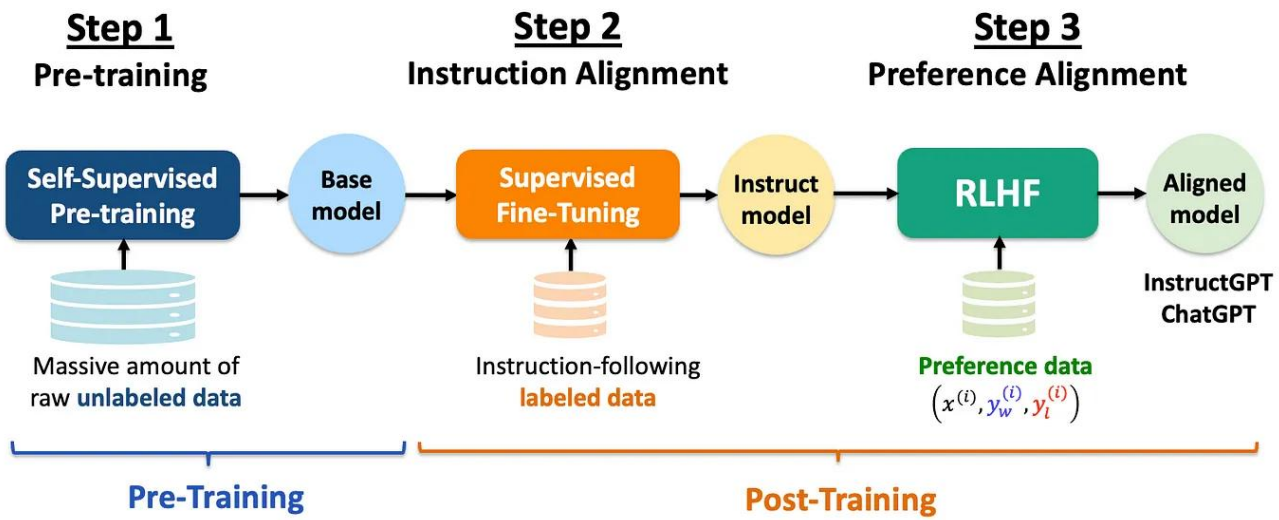
5. Domain-Specific Adaptation (LoRA, QLoRA, fine-tuning)
Organizations often adapt the aligned model for:

- medical tasks
- law
- engineering
- coding
- multilingual tasks

This step is optional and uses small, task-specific datasets.

Pre-Training vs Post Training

- **Pre-training** is the **first and largest** stage of training an LLM. **Pre-training** is the phase where a large language model learns general language patterns by analyzing massive amounts of text. **Outcome is a base/foundational model** that:
  - Knows language well
  - Generates coherent text
  - But **does NOT reliably follow instructions**
  - And is **not aligned** with human values or safety expectations
- **Post-training** refers to *everything done to a base model after pre-training* to make it more useful, safer, and better aligned with human expectations.
- Pre-training teaches a model general language ability, but **post-training transforms a raw model into a helpful assistant**.



https://todatabeyond.substack.com/p/visual-guide-to-llm-preference-tuning

LLM pre-training

- **Pre-training** is the phase where a large language model learns general language patterns by analyzing massive amounts of text.
  Think of it as teaching the model *how* language works before teaching it to perform specific tasks.
- During pre-training, the model learns by predicting missing or next words in sentences, for example:

  "The cat sat on the ___."
- By doing this billions of times, the model discovers grammar, facts, reasoning patterns, relationships between words, and more.

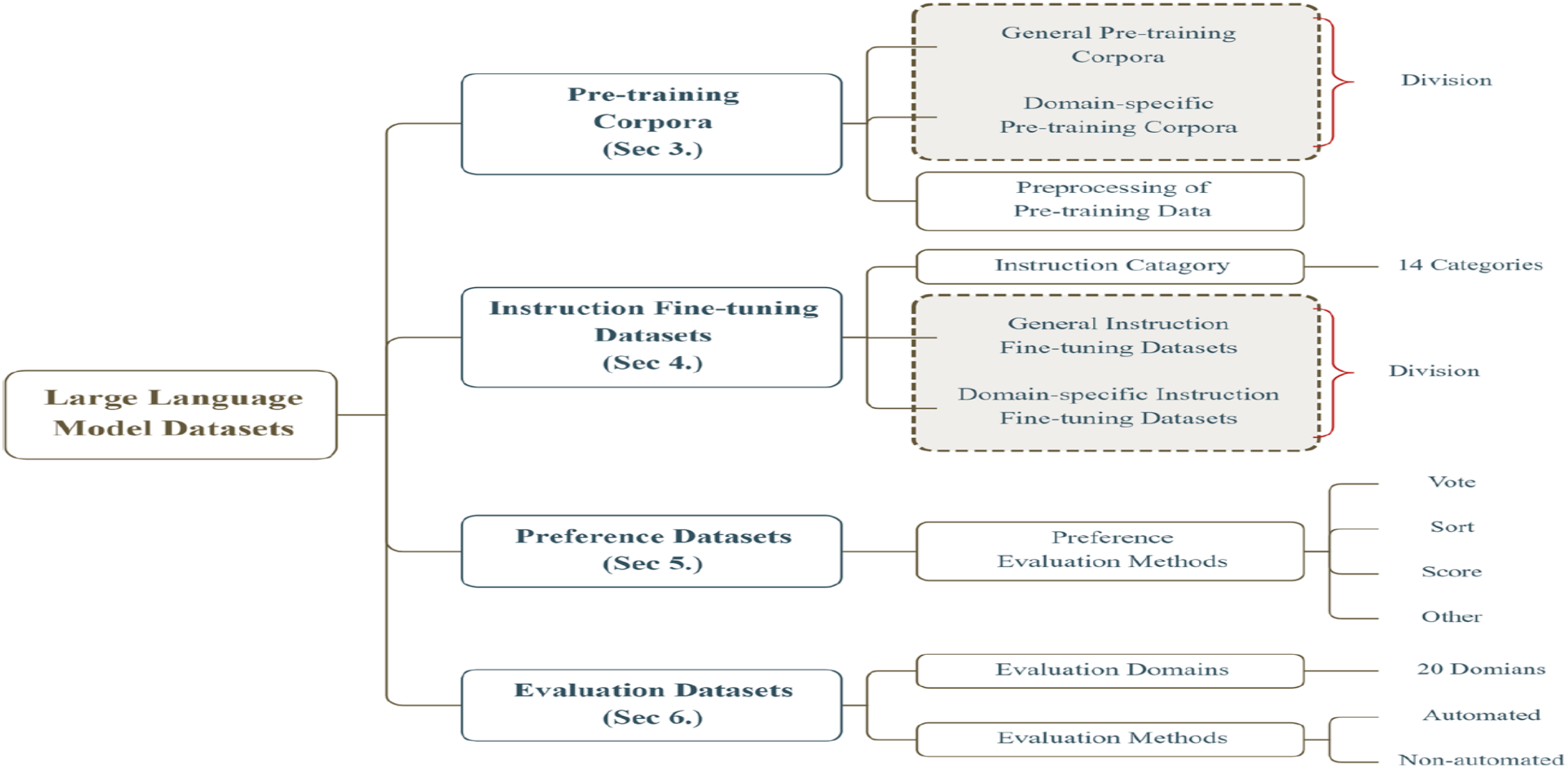
Goals of Pre-Training

- Learn vocabulary and grammar
- Understand context and long-range dependencies in text
- Absorb general world knowledge
- Develop reasoning patterns
- Build a flexible foundation for future task-specific fine-tuning

How It Works (Simplified)

1. **Huge datasets** are collected (books, websites, articles, code, etc.).
2. Text is cleaned and tokenized (split into smaller units).
3. The neural network is trained using self-supervised learning:
    - Next-token prediction
    - Masked-token prediction
4. The model adjusts billions of parameters to minimize prediction error.
5. After pre-training, the model can generate coherent text and understand language broadly.

LLM Datasets



Datasets for large language models: a comprehensive survey [pdf]

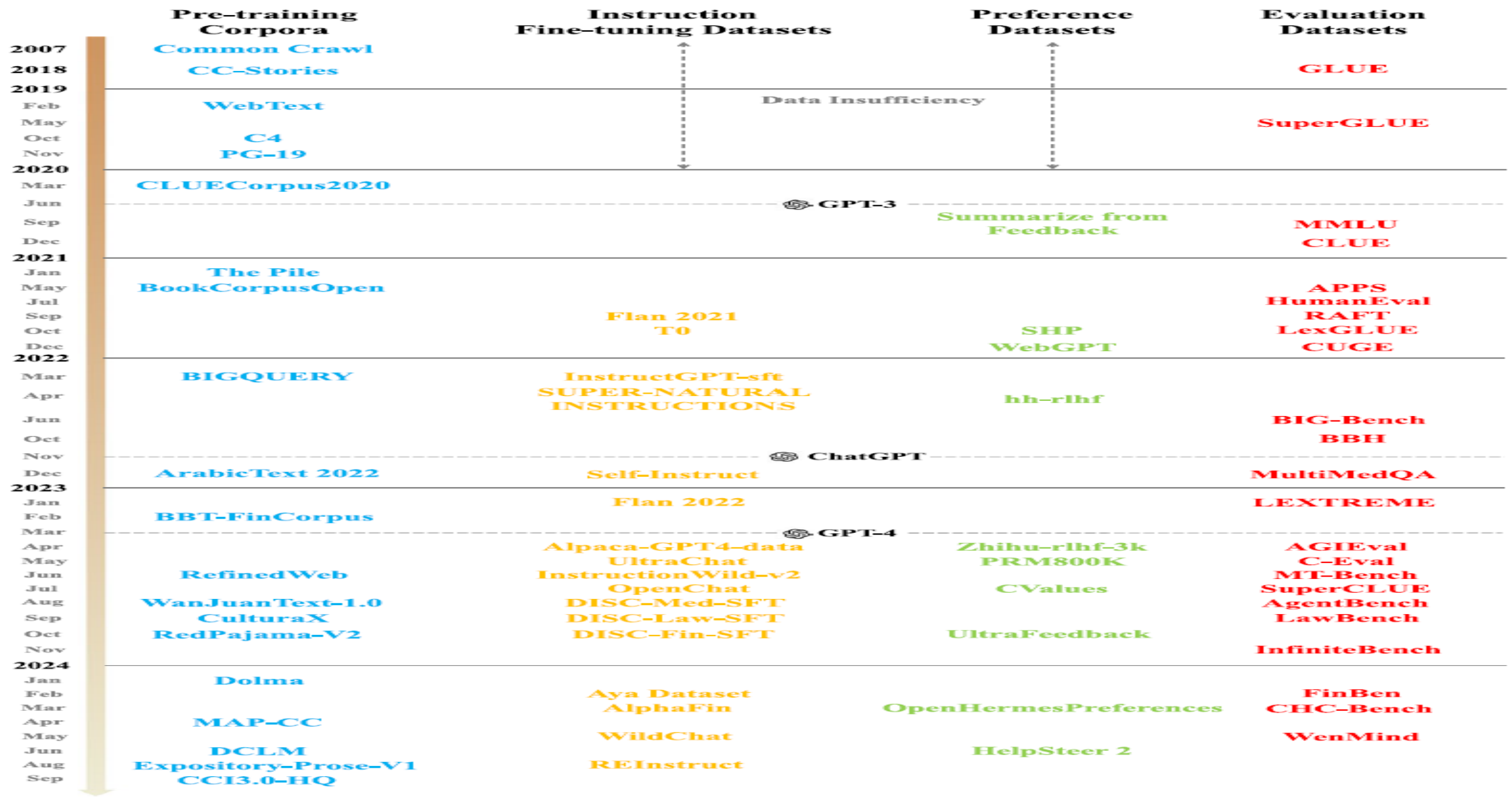


Fig. 2 A timeline of some representative LLM datasets

Datasets for large language models: a comprehensive survey [pdf]

Major Dataset Categories Used in LLM Pre-Training

Large Language Models are pre-trained on **massive, diverse text corpora**. These come from several major categories:

1. Web-Scale Crawls (e.g., Common Crawl, C4, OSCAR)

Web data is the **largest and most common source** for LLM pre-training.

- Provides hundreds of terabytes of raw text from websites, news, blogs, forums, encyclopedias, etc.
- Powers many well-known datasets such as **C4** and **OSCAR**.
- Offers huge diversity but can be noisy, requiring heavy filtering. [apxml.com]

2. Open, Ethical, Large-Scale Datasets (e.g., Common Corpus)

Recent work focuses on legally clean, transparent, open datasets:

- **Common Corpus** → 2 trillion tokens of uncopyrighted or permissively licensed data.
- Contains multilingual text, books, newspapers, scientific articles, legal documents, and code.
- Used by companies such as Anthropic for training open models. [arxiv.org], [builders.mozilla.org]

Major Dataset Categories Used in LLM Pre-Training

3. Books and Long-Form Text

Books provide:

- High-quality, coherent long-form language
- Rich narrative structure and complex syntax
  Public-domain sources include Project Gutenberg; other curated book corpora are used where legally allowed. [apxml.com]

4. Academic and Scientific Text

Includes:

- Scientific papers
- Educational texts
- Technical manuals
  Large datasets exist that include millions of PDFs of academic material. [builders.mozilla.org]

Major Dataset Categories Used in LLM Pre-Training

5. Code Repositories

Essential for models that handle programming tasks.

- Includes data from GitHub and other open-source code repositories
- Helps models learn formal syntax, logic, and structured reasoning [apxml.com]

6. Multilingual Corpora

Modern LLMs rely heavily on multilingual sources:

- Webpages in many languages
- Parallel corpora
- Multilingual open datasets such as OSCAR or the multilingual section of Common Corpus [builders.mozilla.org]

Major Dataset Categories Used in LLM Pre-Training

7. Domain-Specific Pre-Training Datasets

These enrich the model with specialized knowledge:

- **Finance** (e.g., Finance Commons with 1.25M PDFs)
- **Medical**
- **Legal**
- **Math**

  These datasets are often used for continued pre-training (a "second stage"). [deepwiki.com]

8. Curated Public Corpora Lists (Meta-Catalogs)

Repositories like **LLMDataHub** and **Awesome-LLMs-Datasets** organize and index hundreds of pre-training datasets across:

- Webpages
- Books
- Code
- Academic texts
- Social media
- Financial/medical/math domains

  Total pre-training corpora surveyed exceed **774 TB**. [github.com], [link.springer.com], [deepwiki.com]

Summary

Category	Description	Examples / Sources
Web crawls	Largest, most diverse raw text	Common Crawl, C4, OSCAR [apxml.com]
Open ethical datasets	Legally clean, ≥2T tokens	Common Corpus [arxiv.org], [builders.mozilla.org]
Books	Edited, long-form, high quality	Project Gutenberg, public-domain books [apxml.com]
Academic text	Scientific papers, PDFs	Finance Commons, scientific publications [builders.mozilla.org]
Code	Structured programming data	GitHub-derived datasets [apxml.com]
Multilingual corpora	Non-English text	Common Corpus multilingual section, parallel corpora [builders.mozilla.org]
Domain-specific	Specialized corpora	Financial, medical, math sets [deepwiki.com]
Meta-catalogs	Curated lists of datasets	LLMDataHub, Awesome-LLMs-Datasets [github.com], [link.springer.com], [deepwiki.com]

Eight major data categories

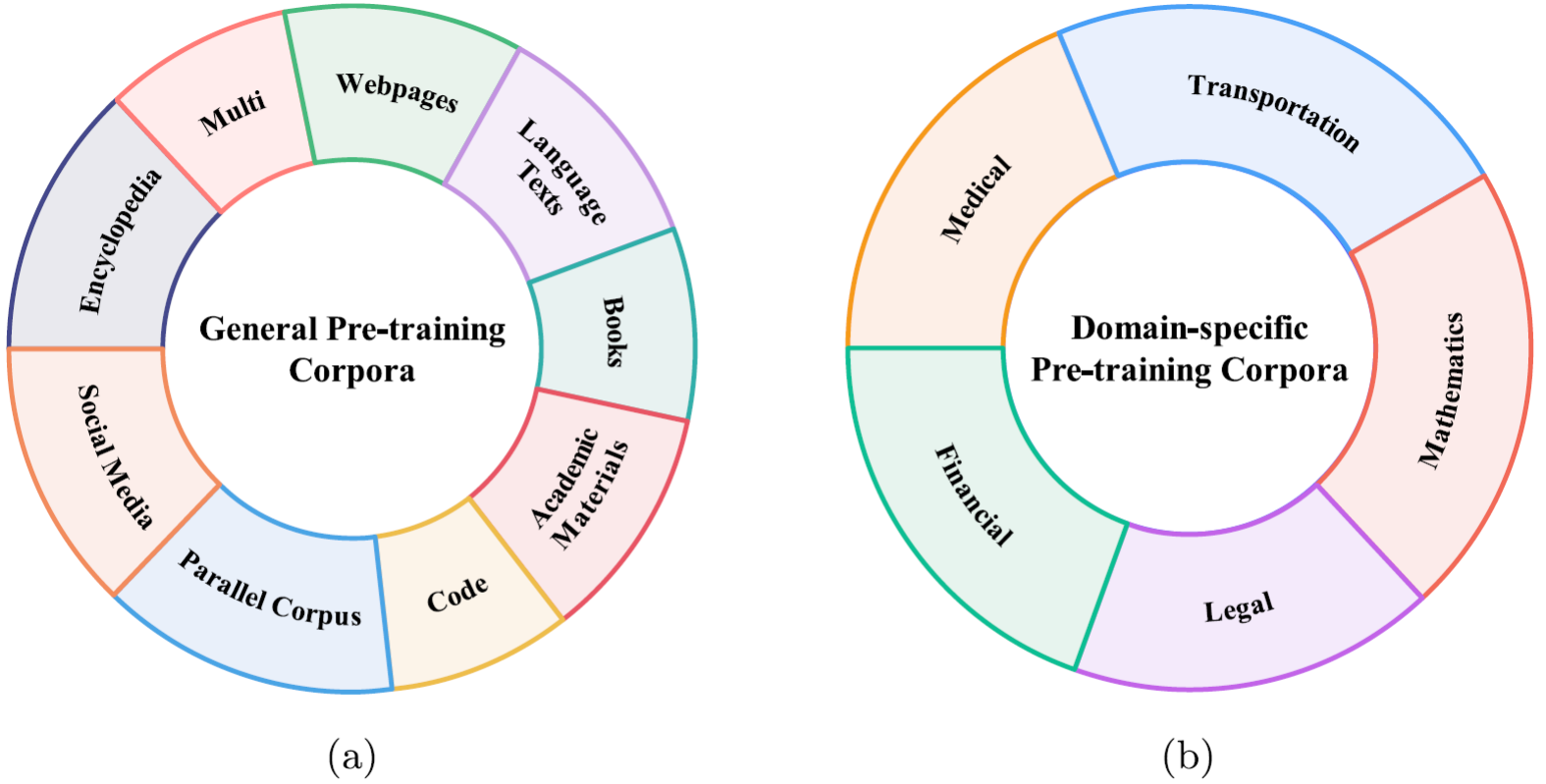


Fig. 3 **(a)** Data categories of the general pre-training corpora; **(b)** Domain categories of the domain-specific pre-training corpora

A **parallel corpus** contains:
- **Source text** (e.g., English)
- **Target text** (e.g., Turkish)
- **Alignment links** showing which sentences correspond across languages

Example:
These pairs form **parallel data**.

English Sentence	Turkish Sentence
The cat is on the table.	Kedi masanın üzerinde.
I like machine learning.	Makine öğrenmesini seviyorum.

Datasets for large language models: a comprehensive survey [pdf]

Core Preprocessing Stages in LLM Data Pipelines

1. Data Ingestion & Loading
Collecting large datasets from web crawls, books, code repositories, and storage systems requires high-throughput, distributed I/O pipelines. [apxml.com]

2. Data Cleaning
Data cleaning is the foundational step in preparing LLM datasets.
Key cleaning operations:

- **Remove HTML markup, boilerplate, ads, navigation text** (common in web crawls) [apxml.com]
- **Remove duplicate data** (exact and near-duplicate lines or documents)
  - Prevents memorization and improves generalization. [thetexttool.com]
- **Fix Unicode inconsistencies** (normalize to NFC form)
  - Ensures that visually identical characters like “café” are treated consistently. [thetexttool.com]
- **Whitespace cleanup**
  - Collapse multiple spaces, remove leading/trailing whitespace, fix inconsistent indentation. [thetexttool.com]
- **Remove unnecessary symbols, emojis (domain-dependent)**
  - For formal domains like legal or academic, emojis add noise. [thetexttool.com]
- **Handle special characters and control codes**
  - Remove zero-width characters, control codes. [thetexttool.com]
- **Language separation and preliminary text cleaning** when processing multilingual text. [developer.nvidia.com]

Core Preprocessing Stages in LLM Data Pipelines

3. Normalization

Standardizing text representation ensures uniform processing.

Common normalization steps:

- **UTF-8 encoding enforcement** [github.com]
- **Standardizing whitespace and special patterns** (e.g., "--") [github.com]
- **Normalize text formats** such as JSON, XML, or MDX for structured datasets. [latitude.so]

4. Sampling & Filtering

Filtering removes irrelevant or low-quality data.

Filtering tasks:

- **Remove very short lines**, empty lines, or excessively long lines. [thetexttool.com]
- **Filter low-quality or noisy text** using heuristic or model-based filters.
  - NVIDIA notes that filtering toxic, low-quality, or PII-containing text is crucial. [developer.nvidia.com]
- **Deduplicate documents and paragraphs** (using MinHash, FAISS, etc.). [latitude.so]
- **Dataset balancing** across domains or languages. [github.com]

Core Preprocessing Stages in LLM Data Pipelines

5. Tokenization / Encoding

Tokenization converts text into the numeric tokens LLMs use.

Common tokenization methods:

- **Byte Pair Encoding (BPE)** – used in GPT-style models. [github.com]
- **WordPiece** – used in BERT-style models. [github.com]
- **Unigram Language Model** – used in SentencePiece; more flexible. [github.com]
- **Byte-level BPE** – handles arbitrary characters, including emojis.
  Used in modern GPT families. [github.com]

Tokenization is one of the **most computationally expensive** steps at scale. [apxml.com]

6. Quality Scoring & Model-Based Filtering

More advanced pipelines include:

- **Heuristic filters** for quality, toxicity, or format.
- **Model-based scoring**, where smaller models rate text quality before inclusion.
  (e.g., used in NVIDIA NeMo Curator workflows) [developer.nvidia.com]

Core Preprocessing Stages in LLM Data Pipelines

7. Deduplication (Exact + Near-Duplicate)

A critical step for:

- Reducing training compute
- Preventing memorization
- Avoiding repeated examples that bias the model [thetexttool.com]

Large-scale deduplication requires distributed processing. [apxml.com]

8. Language Detection & Separation

Especially important for multilingual corpora:

- Identify language per document or paragraph
- Route text to language-specific processing pipelines [developer.nvidia.com]

Core Preprocessing Stages in LLM Data Pipelines

9. Structured Format Conversion

When dealing with logs, JSON, or XML:

- Preserve metadata
- Convert to standardized text fields for LLM consumption [latitude.so]

10. Safety Filtering / PII Removal

Many pipelines include:

- **PII removal** (emails, phone numbers, personal names) - **PII** stands for Personally Identifiable Information
- **Toxicity filtering** (hate, violence, explicit content) [developer.nvidia.com]

This is crucial for both legal compliance and reducing harmful model outputs.

Core Preprocessing Stages in LLM Data Pipelines

Summary: Typical LLM Preprocessing Pipeline

1. **Ingestion** (massive web crawls, books, repositories)
2. **Cleaning** (remove noise, markup, duplicates)
3. **Normalization** (UTF-8, whitespace, text format consistency)
4. **Sampling & Filtering** (quality checks, line/length filtering)
5. **Deduplication** (exact & near-duplicate removal)
6. **Tokenization** (BPE, WordPiece, Unigram, etc.)
7. **Quality scoring** (heuristic or model-based)
8. **Language detection**
9. **Safety filtering** (toxicity, PII)
10. **Packaging for training** (sharding, compression, indexing)

Fig. 7 Flowchart of preprocessing for pre-training corpora

Datasets for large language models: a comprehensive survey [pdf]

Table 1 Summary of *General pre-training corpora* information *Part I*

Corpus	Release time	Size	Public or not	License
ANC	2003-X	–	All	–
Anna's archive	2023-X	641.2 TB	All	–
ArabicText 2022	2022-12	201.9 GB	All	CC-BY-SA—4.0
arXiv	1991-X	–	All	Terms of use for arXiv APIs
Baidu baike	2008-4	–	All	Baidu baike User Agreement
BIGQUERY	2022-3	341.1 GB	Not	Apache—2.0
BNC	1994-X	4124 Texts	All	–
BookCorpusOpen	2021-5	17,868 Books	All	Smashwords terms of service
CCI3.0-HQ	2024-9	500 GB	All	BAAI data usage protocol
CC-Stories	2018-7	31 GB	Not	–
CC100	2020-7	2.5 TB	All	Common crawl terms of use
CLUECorpus2020	2020-3	100 GB	All	MIT
Common crawl	2007-X	–	All	Common crawl terms of use
CulturaX	2023-9	27 TB	All	mC4 and OSCAR
C4	2019-10	12.68 TB	All	ODC-BY and common crawl terms of use
DCLM	2024-6	278.6 TB	All	Common crawl terms of use
Dolma	2024-1	11,519 GB	All	MR agreement
Expository-Prose-V1	2024-8	56 B Tokens	All	MIT
Github	2008-4	–	All	–
MAP-CC	2024-4	840.48 B Tokens	All	CC-BY-NC-ND—4.0

Datasets for large language models: a comprehensive survey [pdf]

Table 2 Summary of *general pre-training corpora* Information *Part II*

Corpus	Language	CM	Category	Source
ANC	EN	HG	Language Texts	American English Texts
Anna's Archive	Multi	HG	Books	Sci-Hub, Library Genesis, Z-Library, etc
ArabicText 2022	AR	HG and CI	Multi	ArabicWeb, OSCAR, CC100, etc
arXiv	EN	HG	Academic materials	ArXiv Preprint
Baidu baike	ZH	HG	Encyclopedia	Encyclopedic Content Data
BIGQUERY	PL	CI	Code	BigQuery
BNC	EN	HG	Language texts	British English Texts
BookCorpusOpen	EN	CI	Books	Toronto Book Corpus
CCI3.0-HQ	ZH	HG	Multi	News, Social Media, Blogs, etc
CC-Stories	EN	CI	Webpages	Common crawl
CC100	Multi (100)	CI	Webpages	Common crawl
CLUECorpus2020	ZH	CI	Webpages	Common crawl
Common Crawl	Multi	HG	Webpages	Web crawler data
CulturaX	Multi (167)	CI	Webpages	mC4, OSCAR
C4	EN	CI	Webpages	Common crawl
DCLM	EN	CI	Webpages	Common crawl
Dolma	EN	HG and CI	Multi	Project Gutenberg, C4, Reddit, etc
Expository-Prose-V1	EN	HG and	Multi	ArXiv, Wikipedia, Gutenberg, etc

Language: "EN" indicates English, "ZH" indicates Chinese, "AR" indicates Arabic, "PL" indicates Programming Language, "Multi" indicates Multilingual, and the number in parentheses indicates the number of languages included. "CM" indicates Construction Methods, where "HG" indicates Human Generated Corpora, "MC" indicates Model Constructed Corpora, and "CI" indicates Collection and Improvement of Existing Corpora.

Datasets for large language models: a comprehensive survey [pdf]

Next: Post-Training

Outcome of pre-training is a base model (also called a *foundation model*)
This model:

- can generate plausible text
- has general knowledge
- can perform many tasks *implicitly*
- but **does not reliably follow instructions**,
  and is **not yet aligned** with human preferences or safety requirements.

Feature	Pre-Training	Post-Training
Goal	Learn general language	Make model useful, safe, and aligned
Data	Massive unstructured text	Curated human or synthetic instructions
Method	Self-supervised (next-token prediction)	Supervised + preference optimization
Output	Base model	Instruction-following, aligned model
Capabilities	Broad knowledge, fluent text	Helpful, safe, accurate task performance
Examples of Methods	Next-token prediction	SFT, RLHF, DPO, domain tuning