Creating a Large Multilingual Dataset for LLM Pre-Training

Guilherme Penedo
guilherme@huggingface.co

Outline

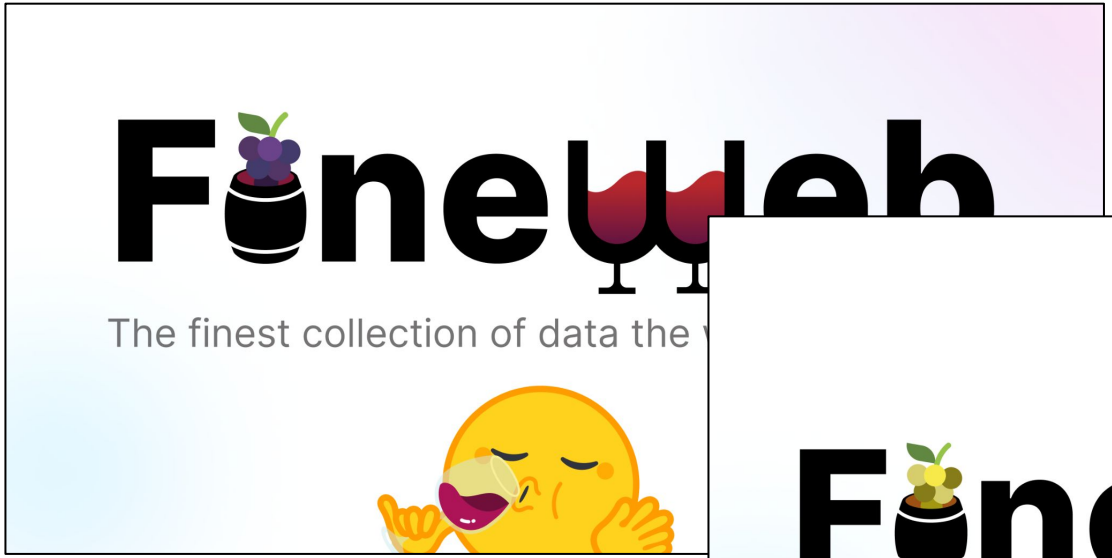
1. Intro
2. Data quality
3. Experiments & Evaluation
4. Separating words

   Short questions break
5. Building FineWeb2
6. Conclusion

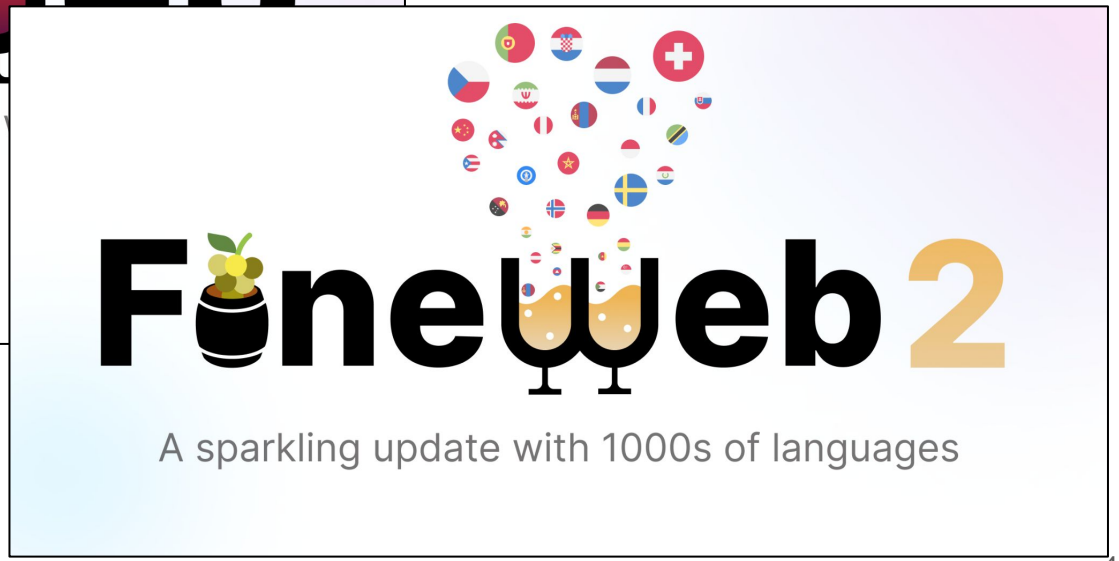
1. Intro

Intro

April 2024



December 2024



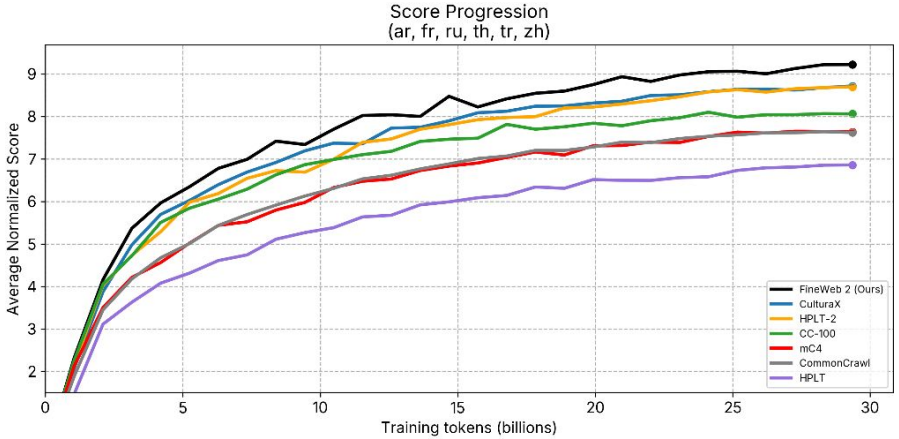
Intro: FineWeb2

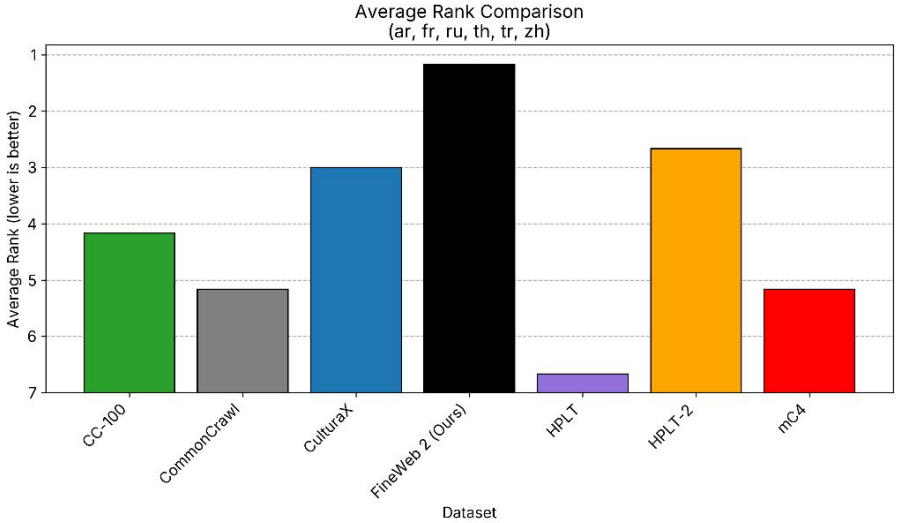
- 1000+ languages
- ~100 CommonCrawl snapshots
- Same data-driven approach as FineWeb

150+ experiments

Model name	Language(s)	Type	Desc	Source data path (For one of the languages)	Tokenized path (For one of the languages)
1p46G-gemma-101b_arabicwords-LANG-29BT-seed-6	ar	Ref-dataset		s3://fineweb-multilingual-v1/ref-datasets/monolingual/ar/101b_arabicwords	s3://fineweb-multilingual-v1/tokenized/ref-datasets/monolingual/ar/101b_arab
1p46G-gemma-arabicweb24-LANG-29BT-seed-6	ar	Ref-dataset		s3://fineweb-multilingual-v1/ref-datasets/monolingual/ar/arabicweb24	s3://fineweb-multilingual-v1/tokenized/ref-datasets/monolingual/ar/arabicweb2
1p46G-gemma-c4q_f-def-LANG-29BT-seed-6	ALL	Filter-exp		s3://fineweb-multilingual-v1/experiments/filtering/c4_quality/en-thr/swahili/output	s3://fineweb-multilingual-v1/tokenized/gemma/filtering/c4_quality/en-thr/sw/
1p46G-gemma-cc-100-LANG-29BT-seed-6	ar,fr,ru,th,tr,zh	Ref-dataset		s3://fineweb-multilingual-v1/ref-datasets/cc-100/zh-hans,s3://fineweb-multilingual-v1/ref-datasets/cc-100/zh-hant	s3://fineweb-multilingual-v1/tokenized/gemma/cc-100/chinese
1p46G-gemma-clean_wikipedia_q2-LANG-29BT-seed-6	ALL	Filter-exp	*some combination of thresholds computed with the new clean wikipedia*	s3://fineweb-multilingual-v1/experiments/new_filtering_exps/clean_wikipedia_q2/rus_Cyrl	s3://fineweb-multilingual-v1/tokenized/nfilexps/clean_wikipedia_q2/rus
1p46G-gemma-clean_wikipedia-LANG-29BT-seed-6	ALL	Filter-exp	some combination of thresholds computed with the new clean wikipedia	s3://fineweb-multilingual-v1/experiments/new_filtering_exps/clean_wikipedia/cmn_Hani	s3://fineweb-multilingual-v1/tokenized/nfilexps/clean_wikipedia/cmn
1p46G-gemma-commoncrawl-LANG-29BT-seed-3	ar,fr,hi,ru,th,tr,zh	Ref-dataset	Seed 3 & 4 sampling/shuffling, diff model seed	s3://fineweb-multilingual-v1/glotlid/cmn,s3://fineweb-multilingual-v1/glotlid/hak,s3://fineweb-multilingual-v1/glotlid/lzh,s3://fineweb-multilingual-v1/glotlid/nan,s3://fineweb-multilingual-v1/glotlid/wuu,s3://fineweb-multilingual-v1/glotlid/yue	s3://fineweb-multilingual-v1/tokenized/gemma/commoncrawl-2seed/chinese
1p46G-gemma-commoncrawl-LANG-29BT-seed-4	ar,fr,hi,ru,th,tr,zh	Ref-dataset	Seed 3 & 4 sampling/shuffling, diff model seed	s3://fineweb-multilingual-v1/glotlid/acf,s3://fineweb-multilingual-v1/glotlid/crs,s3://fineweb-multilingual-v1/glotlid/fra,s3://fineweb-multilingual-v1/glotlid/gcf,s3://fineweb-multilingual-v1/glotlid/gcr,s3://fineweb-multilingual-v1/glotlid/mfe,s3://fineweb-multilingual-v1/glotlid/nrf,s3://fineweb-multilingual-v1/glotlid/pcd,s3://fineweb-multilingual-v1/glotlid/rcf	s3://fineweb-multilingual-v1/tokenized/gemma/commoncrawl-2seed/french
1p46G-gemma-commoncrawl-LANG-29BT-seed-5	ALL	Ref-dataset	Seed 5 & 6 sampling/shuffling, diff model seed	s3://fineweb-multilingual-v1/glotlid/swc,s3://fineweb-multilingual-v1/glotlid/swh	s3://fineweb-multilingual-v1/tokenized/gemma/commoncrawl/swahili
1p46G-gemma-commoncrawl-LANG-29BT-seed-6	ALL	Ref-dataset	Seed 5 & 6 sampling/shuffling, diff model seed	s3://fineweb-multilingual-v1/glotlid/tel	s3://fineweb-multilingual-v1/tokenized/gemma/commoncrawl/telugu
1p46G-gemma-commoncrawl-LANG-29BT-seed-7	sw,te	Ref-dataset	Seed 5,6,7&8 shuffling, diff model seed (te,sw only)	s3://fineweb-multilingual-v1/glotlid/swc,s3://fineweb-multilingual-v1/glotlid/swh	s3://fineweb-multilingual-v1/tokenized/gemma/commoncrawl/swahili
1p46G-gemma-commoncrawl-LANG-29BT-seed-8	sw,te	Ref-dataset	Seed 5,6,7&8 shuffling, diff model seed (te,sw only)	s3://fineweb-multilingual-v1/glotlid/tel	s3://fineweb-multilingual-v1/tokenized/gemma/commoncrawl/telugu
1p46G-gemma-croissant-LANG-29BT-seed-6	fr	Ref-dataset		s3://fineweb-multilingual-v1/ref-datasets/monolingual/fr/croissant	s3://fineweb-multilingual-v1/tokenized/ref-datasets/monolingual/fr/croissant/
1p46G-gemma-culturax-LANG-29BT-seed-6	ar,fr,hi,ru,th,tr,zh	Ref-dataset		s3://fineweb-multilingual-v1/ref-datasets/culturax/ru	s3://fineweb-multilingual-v1/tokenized/gemma/culturax/russian
1p46G-gemma-dedup-rehydr-LANG-29BT-seed-6	sw	Interm-step	rehydrated swahili of dedup only	s3://fineweb-multilingual-v1/full-pipeline/dedup-v2/swh_Latn/output	s3://fineweb-multilingual-v1/tokenized/full-pipeline/dedup-v2-rehydr/swh
1p46G-gemma-defi-extract-LANG-29BT-seed-6	ALL	Interm-step	RELEASE Extract after release version LID + dedup (NO FILTERING)	s3://fineweb-multilingual-v1/experiments/dedup_filtering/extract/tha_Thai	s3://fineweb-multilingual-v1/tokenized/dedupfilt/extract/tha
1p46G-gemma-defi-glotcorp10td-LANG-29BT-seed-6	ALL	Filter-exp		s3://fineweb-multilingual-v1/experiments/dedup_filtering/glotcorp10td//swh_Latn	s3://fineweb-multilingual-v1/tokenized/dedupfilt/glotcorp10td/swh
1p46G-gemma-defi-glotcorpog-LANG-29BT-seed-6	ALL	Filter-exp		s3://fineweb-multilingual-v1/experiments/dedup_filtering/glotcorpog/tur_Latn	s3://fineweb-multilingual-v1/tokenized/dedupfilt/glotcorpog/tur
1p46G-gemma-defi-glotcorpupd-LANG-29BT-seed-6	ALL	Filter-exp		s3://fineweb-multilingual-v1/experiments/dedup_filtering/glotcorpupd//tel_Telu	s3://fineweb-multilingual-v1/tokenized/dedupfilt/glotcorpupd/tel
1p46G-gemma-defi-glotcorpupd2-LANG-29BT-seed-6	ALL	Filter-exp		s3://fineweb-multilingual-v1/experiments/dedup_filtering/glotcorpupd2//hin_Deva	s3://fineweb-multilingual-v1/tokenized/dedupfilt/glotcorpupd2/hin
1p46G-gemma-defi-rehydr-LANG-29BT-seed-6	ALL	Interm-step	final filtering rehydrated (sans charsdup - data removed by chars dup rule that we readded) old upsampling weights	s3://fineweb-multilingual-v1/full-pipeline/filtering-v2/output/arb_Arab/train	s3://fineweb-multilingual-v1/tokenized/full-pipeline/filtering-v2-rehydr/arb
1p46G-gemma-defi-rehydrfix-LANG-29BT-seed-6	ar,fr,hi,ru,te,th,tr,zh	Interm-step	RELEASE final filtering rehydrated with charsdup data	s3://fineweb-multilingual-v1/full-pipeline/filtering-v2/output/fra_Latn/train,s3://fineweb-multilingual-v1/full-pipeline/filtering-v2/output/fra_Latn/train-chardups	s3://fineweb-multilingual-v1/tokenized/full-pipeline/filtering-v2-rehydrfix/fra

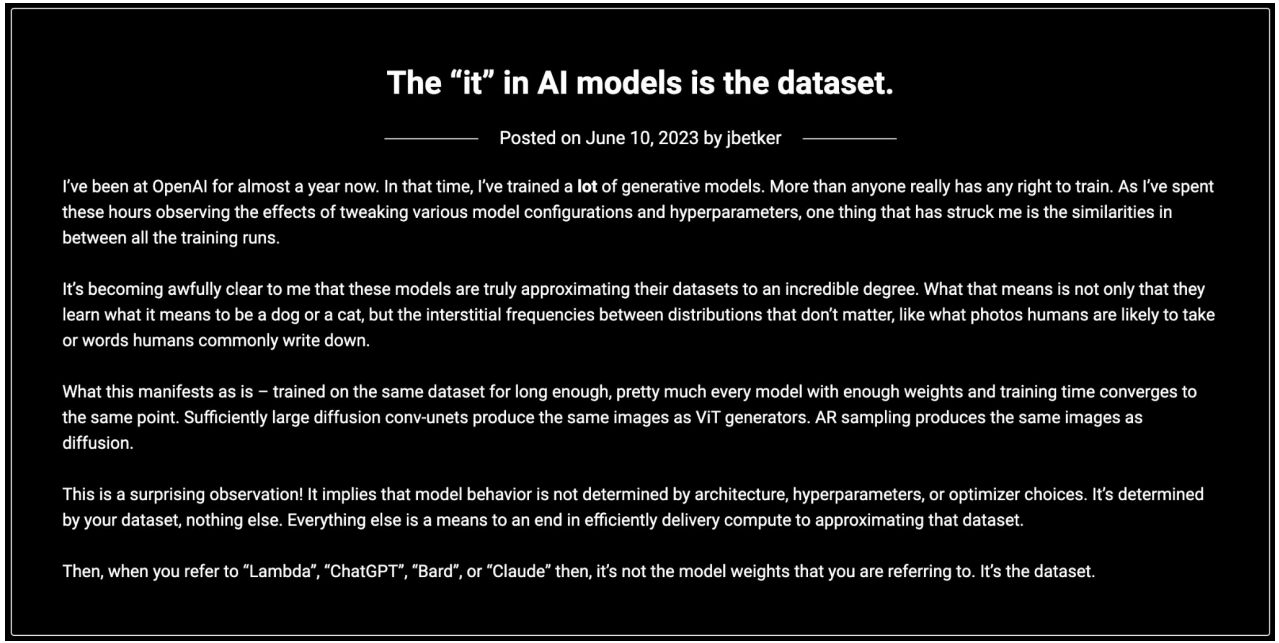
Intro: FineWeb2





2. Data quality

Data quality



The “it” in AI models is the dataset.

Posted on June 10, 2023 by jbetker

I’ve been at OpenAI for almost a year now. In that time, I’ve trained a **lot** of generative models. More than anyone really has any right to train. As I’ve spent these hours observing the effects of tweaking various model configurations and hyperparameters, one thing that has struck me is the similarities in between all the training runs.

It’s becoming awfully clear to me that these models are truly approximating their datasets to an incredible degree. What that means is not only that they learn what it means to be a dog or a cat, but the interstitial frequencies between distributions that don’t matter, like what photos humans are likely to take or words humans commonly write down.

What this manifests as is – trained on the same dataset for long enough, pretty much every model with enough weights and training time converges to the same point. Sufficiently large diffusion conv-unets produce the same images as ViT generators. AR sampling produces the same images as diffusion.

This is a surprising observation! It implies that model behavior is not determined by architecture, hyperparameters, or optimizer choices. It’s determined by your dataset, nothing else. Everything else is a means to an end in efficiently delivery compute to approximating that dataset.

Then, when you refer to “Lambda”, “ChatGPT”, “Bard”, or “Claude” then, it’s not the model weights that you are referring to. It’s the dataset.

https://nonint.com/2023/06/10/the-it-in-ai-models-is-the-dataset/

Data quality: Pretraining

- Base models: general purpose models
- Maximal coverage/diversity
- Massive quantities of text

Download & process “the internet”
(aka CommonCrawl)

Data quality: How to evaluate?

- Hard to define
- Manual data inspection
  - Top domains
  - Kept/removed documents
  - Important but not scalable
- Gold standard perplexity
  - Biases
  - Not always correlated model perf
- **Train (small) models!**

Data quality: Train models!

Model **A**

- **X** params
- Arch **Y**
- Tokenizer **Z**
- **N** tokens
- Trained on dataset **A**

Model **B**

- **X** params
- Arch **Y**
- Tokenizer **Z**
- **N** tokens
- Trained on dataset **B**

Evaluate & compare:

Score **B** > Score **A** ⇒ Dataset **B** > Dataset **A**

3. Experiments & Evaluation

Experiment setup

- Monolingual models
- Impossible to train for all languages :(
- Train on **9 canary languages**
- Diverse in:
  - Family
  - Script
  - Resource availability
- (should have benchmarks)

9 canary languages

Language	Code	Family	Script	Resource availability
Arabic	ar / arb	Afro-Asiatic	Arabic	Medium
Chinese	zh / cmn	Sino-Tibetan	Han	High
French	fr / fra	Indo-European (Italic)	Latin	High
Hindi	hi / hin	Indo-European (Indo-Iranian)	Devanagari	Medium
Russian	ru / rus	Indo-European (Balto-Slavic)	Cyrillic	High
Swahili	sw / swh	Niger-Congo	Latin	Low
Telugu	te / tel	Dravidian	Telugu	Low
Thai	th / tha	Kra-Dai	Thai	Medium
Turkish	tr / tur	Turkic	Latin	Medium

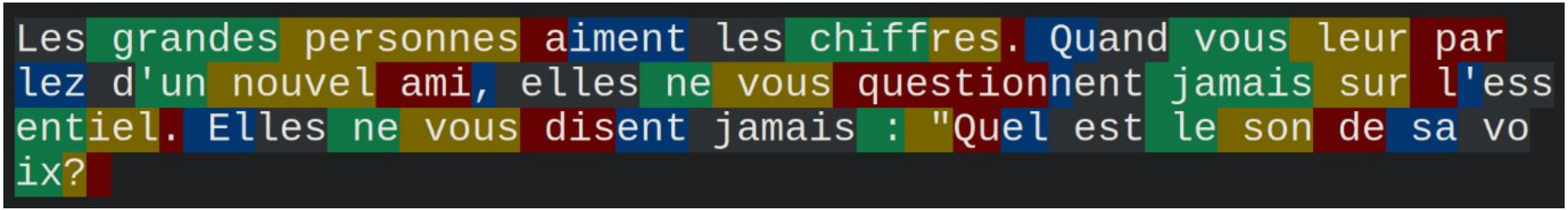
Tokenizer

- New or off the shelf?
  - *Off the shelf*
- 1 per language or 1 for all?
  - *1 for all simplifies the setup*
- Vocab size?
  - *Trade offs for model size*
- How to compare?
  - *Subword fertility*
  - *Proportion continued words*

Tokenizer: metrics

Subword fertility (sf)
Tokens per real word. How aggressively a tokenizer splits. (1+)
1 = tokenizer vocab contains all words

Proportion continued words (pcw)
Real text words encoded w/ 2 or more tokens. How often a tokenizer splits. (0-1)
0=never splits; 1=always splits



Rust et al. (2020) “How Good is Your Tokenizer? On the Monolingual Performance of Multilingual Language Models”

(Lower is better)

Tokenizer - metrics on Wikipedia

		English		Chinese		French		Russian		Turkish		Arabic		Thai		Hindi		Swahili		Telugu		Average		Maximum	
tokenizer	#vocab	sf	pcw	sf	pcw	sf	pcw	sf	pcw	sf	pcw	sf	pcw	sf	pcw	sf	pcw	sf	pcw	sf	pcw	sf	pcw	sf	pcw
Mistral v3	32768	1.45	0.23	3.03	0.95	1.69	0.40	2.42	0.59	3.18	0.74	4.76	0.92	4.87	0.93	4.99	0.91	2.30	0.63	9.83	0.79	4.12	0.76	9.83	0.95
Phi3	100352	1.40	0.28	2.30	0.58	1.74	0.47	2.99	0.66	2.63	0.70	3.72	0.86	3.80	0.85	4.60	0.90	2.09	0.62	10.11	0.76	3.78	0.71	10.11	0.90
Command-R	255000	1.35	0.22	1.35	0.25	1.50	0.35	1.99	**0.56**	2.13	**0.64**	2.16	0.68	4.01	0.87	3.39	0.80	1.95	0.59	9.74	0.78	3.14	0.61	9.74	0.87
Qwen2.5	151643	*1.47*	*0.29*	*1.44*	*0.31*	*1.76*	*0.47*	*2.50*	*0.64*	*2.55*	*0.70*	*2.23*	*0.67*	*2.44*	*0.64*	*3.98*	*0.86*	*2.16*	*0.63*	*8.41*	*0.77*	*3.05*	*0.63*	*8.41*	*0.86*
Llama3	128000	1.40	0.28	1.60	0.43	1.73	0.47	2.34	0.62	2.32	0.68	2.32	0.74	2.18	0.66	2.71	0.81	2.07	0.62	10.11	0.76	3.04	0.64	10.11	0.81
bigsci-bloom	250680	1.42	0.31	**1.29**	**0.23**	**1.49**	0.35	2.86	0.63	2.59	0.67	**1.86**	**0.60**	3.96	0.86	**1.59**	**0.39**	**1.72**	**0.52**	**2.10**	**0.59**	2.16	**0.54**	3.96	0.86
Gemma	256000	**1.31**	**0.19**	1.43	0.32	1.50	**0.34**	**2.05**	0.57	**2.22**	0.66	2.19	0.69	**1.92**	**0.46**	2.22	0.60	1.84	0.53	3.51	0.74	**2.10**	0.55	**3.51**	**0.74**
~~mT5~~	~~250100~~	~~1.52~~	~~0.45~~	~~2.29~~	~~0.91~~	~~1.71~~	~~0.55~~	~~1.96~~	~~0.73~~	~~1.99~~	~~0.73~~	~~2.10~~	~~0.79~~	~~1.99~~	~~0.68~~	~~2.02~~	~~0.69~~	~~1.78~~	~~0.62~~	~~2.44~~	~~0.86~~	~~2.03~~	~~0.73~~	~~2.44~~	~~0.91~~
~~XGLM~~	~~256008~~	~~1.34~~	~~0.28~~	~~2.21~~	~~0.82~~	~~1.45~~	~~0.35~~	~~1.68~~	~~0.50~~	~~1.72~~	~~0.53~~	~~1.72~~	~~0.52~~	~~1.78~~	~~0.53~~	~~1.52~~	~~0.33~~	~~1.54~~	~~0.42~~	~~2.24~~	~~0.69~~	~~1.76~~	~~0.52~~	~~2.24~~	~~0.82~~
~~Cohere-ML~~	~~501153~~	~~1.17~~	~~0.12~~	~~1.62~~	~~0.49~~	~~1.27~~	~~0.18~~	~~1.55~~	~~0.30~~	~~1.51~~	~~0.33~~	~~1.53~~	~~0.31~~	~~1.99~~	~~0.51~~	~~1.32~~	~~0.17~~	~~1.28~~	~~0.19~~	~~1.67~~	~~0.27~~	~~1.52~~	~~0.31~~	~~1.99~~	~~0.51~~

Issues with “unk”, preserving \s

Too big

Final ablation setup

- Similar to original FineWeb
- Llama architecture
- 14 hidden layers (24 originally)
- 32 attention heads
- 2048 sequence length
- Tied embeddings

~1.46B parameters

Chinchilla optimal at ~29BT

Selecting tasks

Easy for English:

- Well established benchmarks (MMLU, Hellaswag, etc)
- Widely used
- Supported by all eval frameworks

For non-English:

- Machine-translated (translationese)
- Not widely validated/hard to find
- Specially worse for low-res

Assumptions

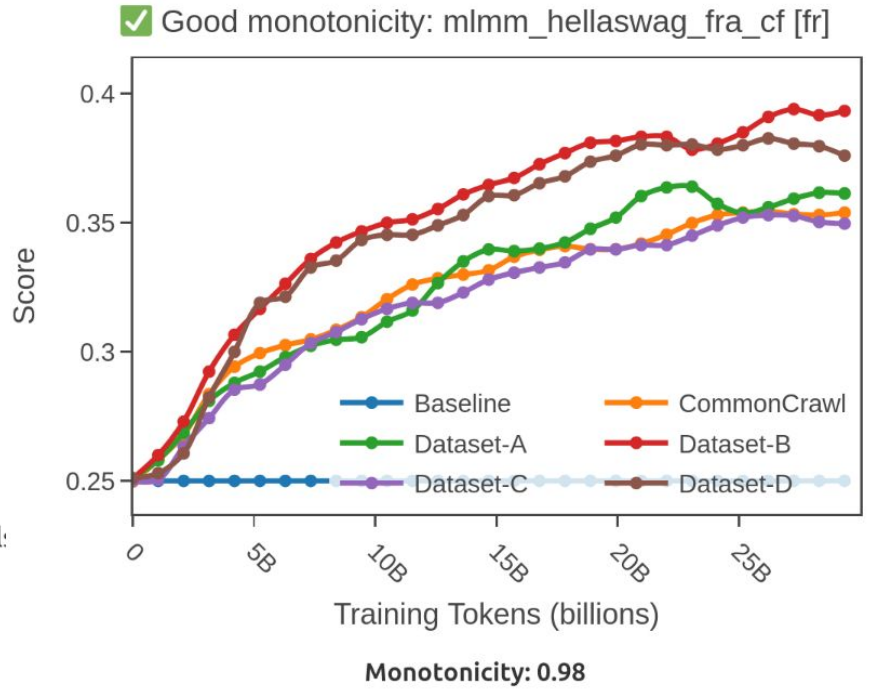
- We have **N** “reference” datasets
  - mC4
  - CC-100
  - CulturaX
  - HPLT
  - “raw/dirty” CommonCrawl
- We trained **N** models on these datasets
- We evaluated checkpoints from each model on many different tasks

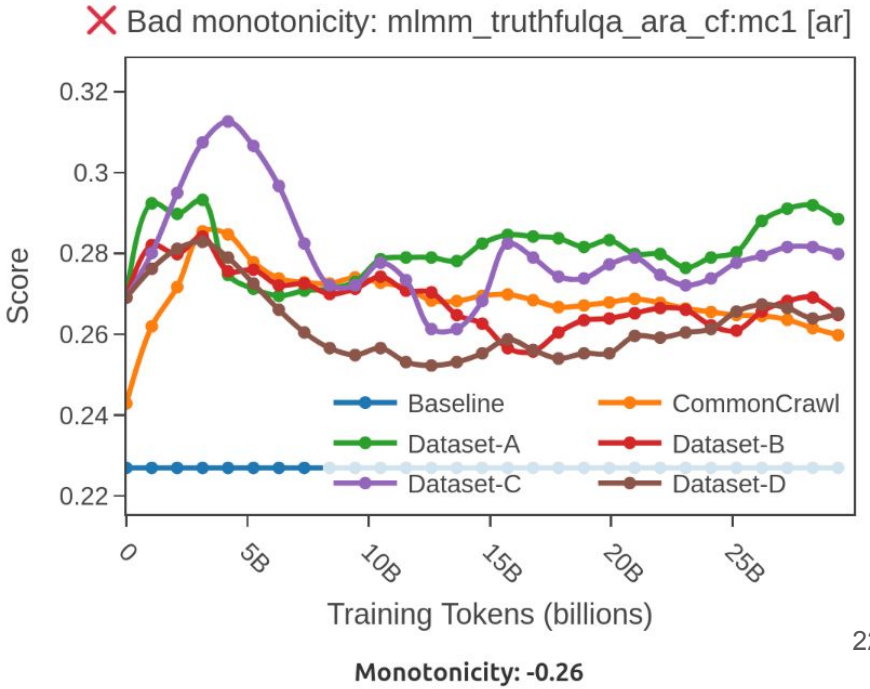
How do we choose **high-signal** tasks for pretraining?

High-signal: monotonicity

Rationale: We should see learning as training progresses

Measure: *Spearman rank correlation* between steps and score

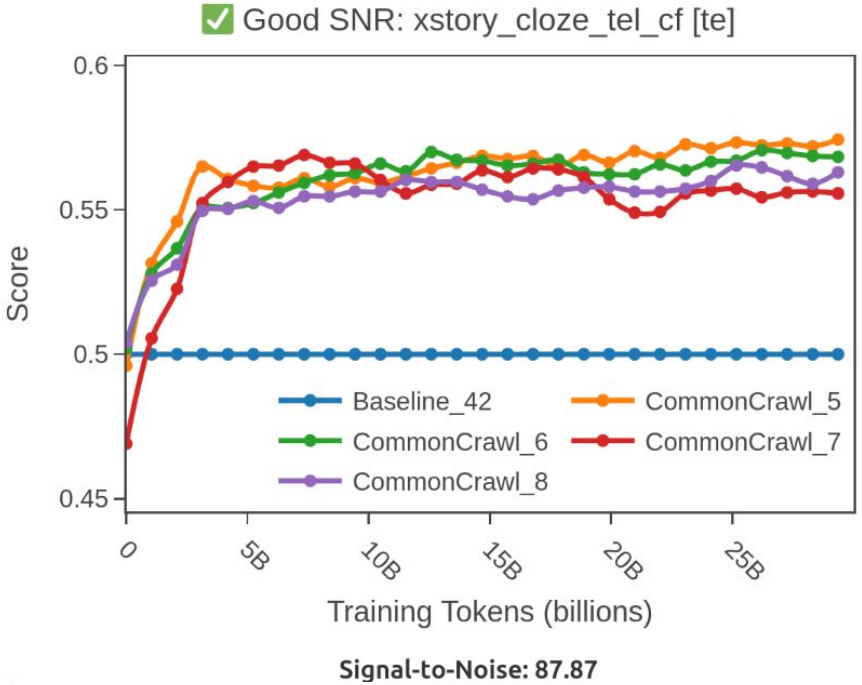


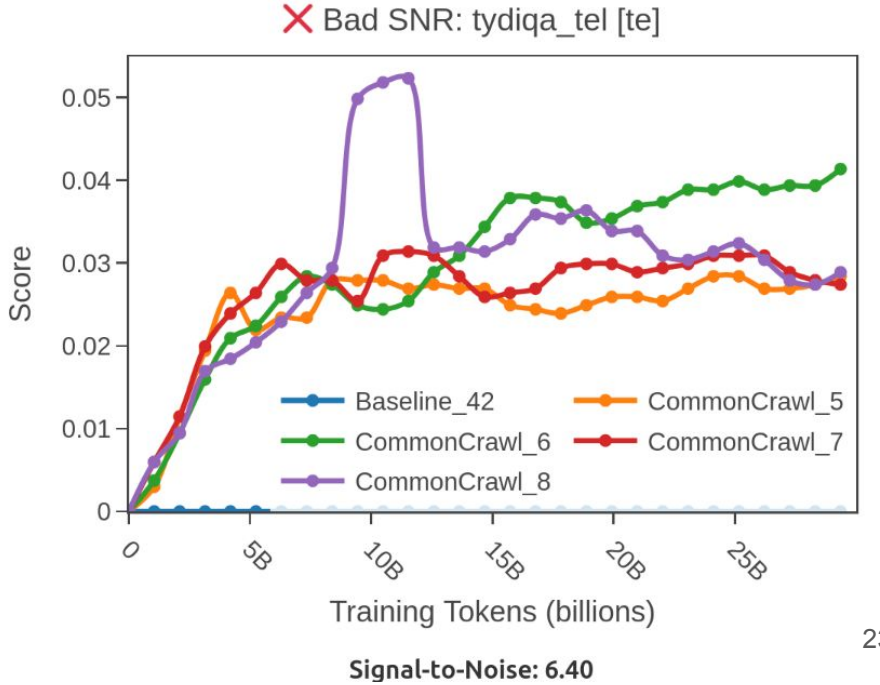


High-signal: low noise

Rationale: Score differences should not be caused by evaluation noise

Measure: *SNR = (avg score / std_dev);* with std_dev coming from diff seeds of “noisy” data

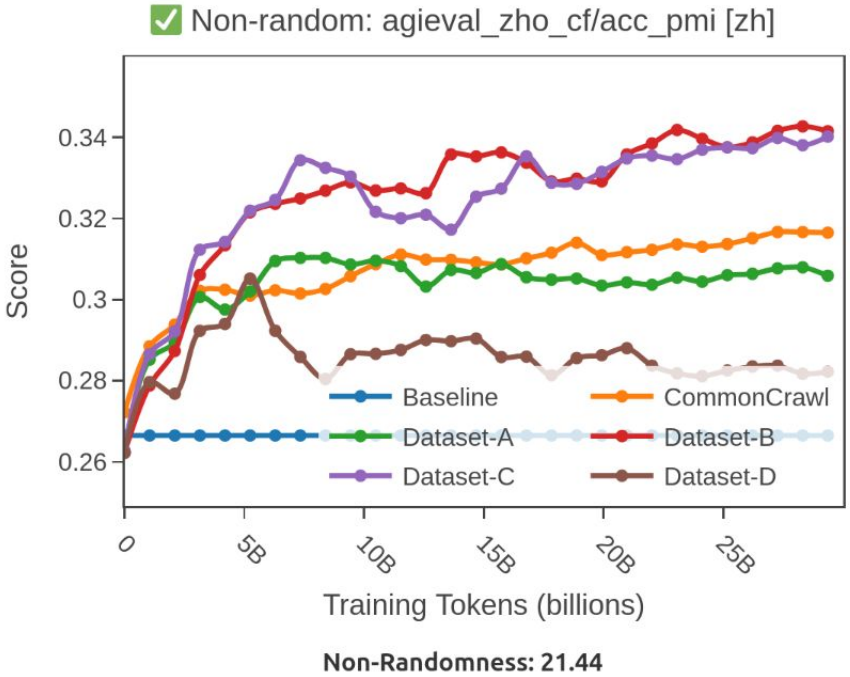


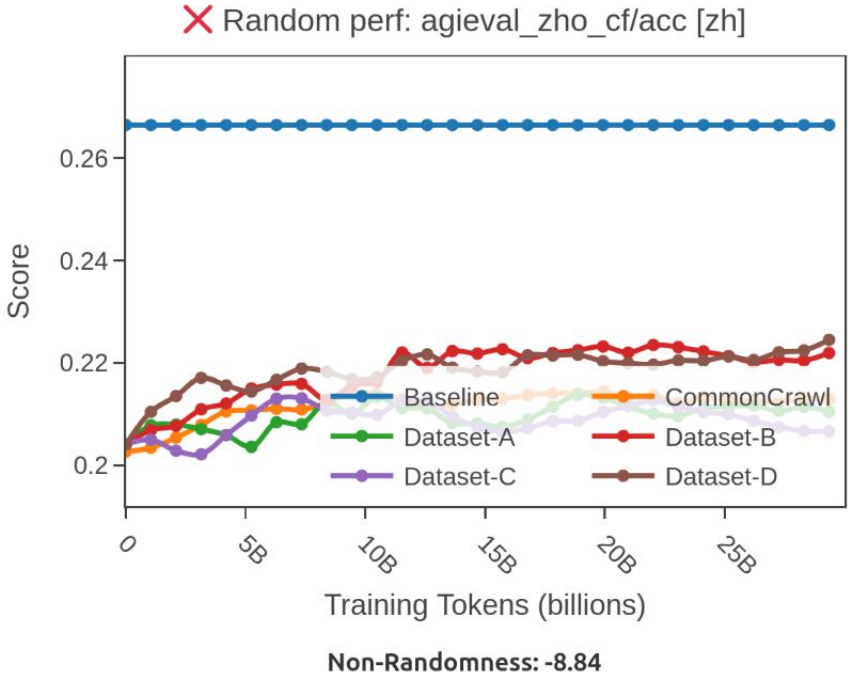


High-signal: above random

Rationale: Can not conclude anything if the model has random performance [for pretraining ablations!]

Measure: *Max distance to RB in std_dev;* with std_dev coming from diff seeds of “noisy” data

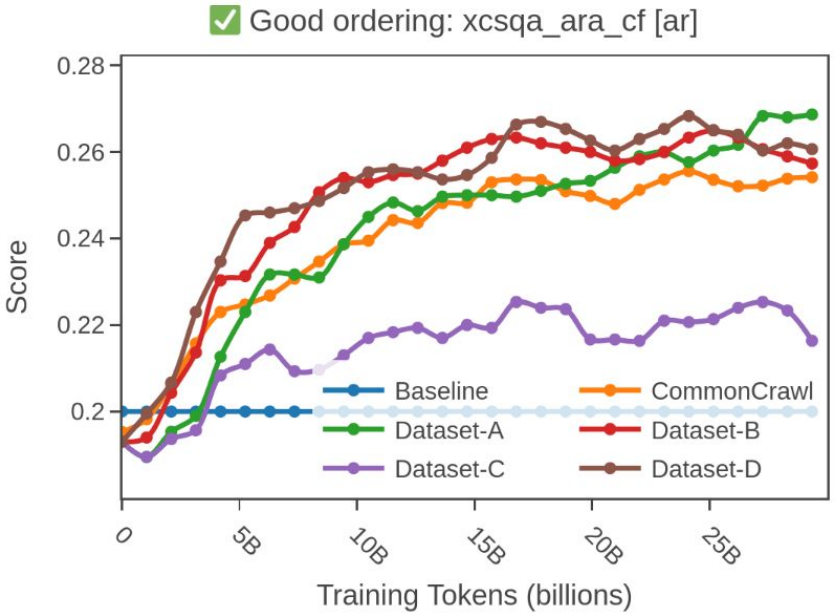


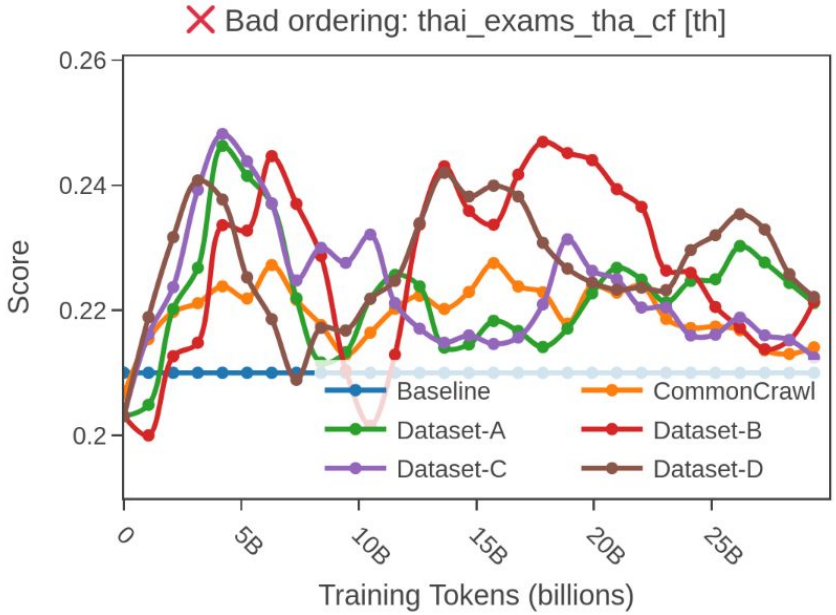


High-signal: ordering consistency

Rationale: We want to generalize to larger scales, pre-condition for that is stable ordering at the experiment scale

Measure: *Kendall-tau for every consecutive step pair*





Evaluation quirks: CF vs MCF

Multiple choice formulation (MCF)
Question: What is the median international income as of 2020?
A. $300
B. $1,000
C. $10,000
D. $30,000
Answer:

Targets: “ A.”, “ B.”, “ C.”, “ D.”

Cloze formulation (CF) ←
Question: What is the median international income as of 2020?
Answer:

Targets: “ $300”, “ $1,000”, “ $10,000”, “ $30,000”

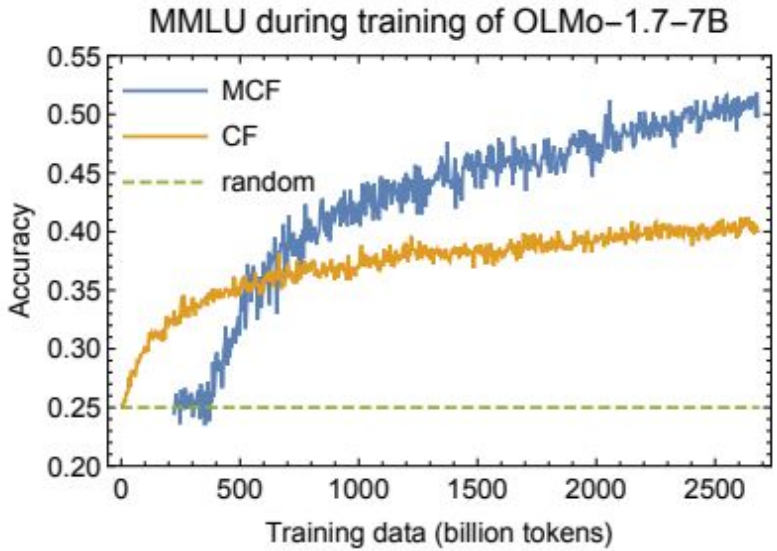


Figure 1: Performance on MMLU validation set during the training of OLMo-1.7-7B model.

Groeneveld et al. (2024) “OLMo: Accelerating the Science of Language Models”

Normalizing logprobs

Account for different completion lengths/baseline logprobs

- **raw**
- **_norm** (character length normalization)
- **_norm_token** (nb of tokens normalization)
- **_norm_pmi** - use unconstrained logprobs to normalize, i.e., logprobs of `Answer: (completion)` without the question

Normalizing logprobs: when to use what

Account for different completion lengths/baseline logprobs

- **raw:** all answers are a single token
- **_norm**: often used when comparing models with diff tokenizers
- **_norm_token**: comparing models with the same tokenizer ←
- **_norm_pmi** - usually for weird or “unlikely” answers, e.g., “pink giraffe with green stripes” - will usually be unlikely

Evaluation signal: using continuous metrics

$$
\begin{aligned}
&\mathrm{Corr}\left(\mathrm{Compute}, \log p_\theta^{\mathrm{Vocab}}(\mathrm{Correct\ Choice})\right) \\
&\quad \geq \mathrm{Corr}\left(\mathrm{Compute}, p_\theta^{\mathrm{Vocab}}(\mathrm{Correct\ Choice})\right) \\
&\quad > \mathrm{Corr}\left(\mathrm{Compute}, p_\theta^{\mathrm{Choices}}(\mathrm{Correct\ Choice})\right) \\
&\quad \geq \mathrm{Corr}\left(\mathrm{Compute}, \texttt{Brier Score}\right) \\
&\quad > \mathrm{Corr}\left(\mathrm{Compute}, \texttt{Accuracy}\right)
\end{aligned}
$$

Schaeffer et al. (2024) “Why Has Predicting Downstream Capabilities of Frontier AI Models with Scale Remained Elusive”

Accuracy vs Probability

- Accuracy is closer to actual tasks people do
- Easier to understand conceptually
- Clear meaning and well defined baseline (1/#answers)
- Noisier
- Normalization greatly affects results

- Relatively abstract not very realistic format
-
- Somewhat obscure (is 50% good (?))
- Random baseline more abstract
-
- Usually very clean (monotonic etc)
- Normalization doesn’t change *as much*
- Could overfit on specific samples

Evaluation: task diversity

- **Reading comprehension (RC)**: Questions based on context
- **General knowledge (GK)**: Questions about facts without added context.
- **Natural Language Understanding (NLU)**: Semantics of provided input.
- **Common-sense reasoning (RES)**: Simple reasoning requiring embodied knowledge.
- **Generative tasks**: Ability to generate text in the target language without the "help" of multiple choice options

Generative tasks metrics

Exact

- Full
- Prefix
- Suffix

F1 ←

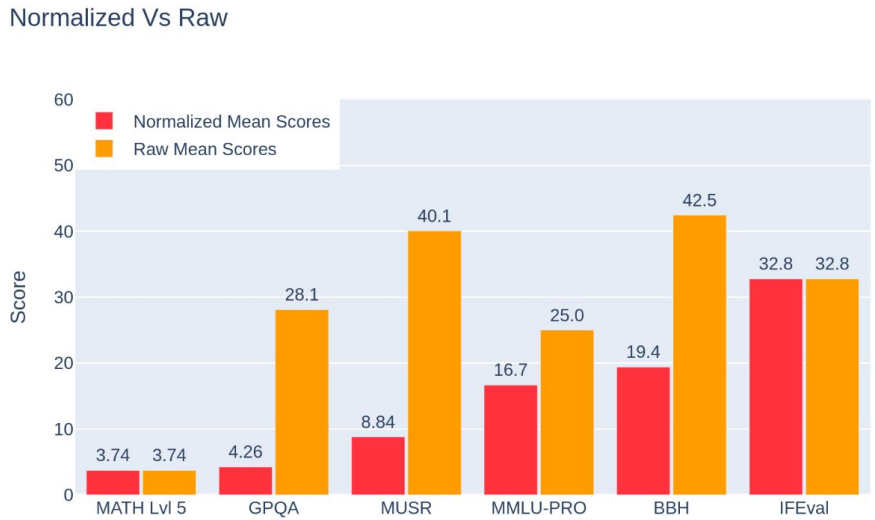
After all that… We get **96 benchmarks** across our 9 languages

To read more
FineTasks: Finding signal in a haystack of 200+ multilingual tasks
https://huggingface.co/spaces/HuggingFaceFW/blogpost-fine-tasks

Aggregating scores from all tasks

- **Simple average** [FineWeb] - not great
- **Rescale** with RB [OpenLLMLeaderboard]
- Our approach: **Z-Score**

$$z = \frac{x - \mu}{\sigma}$$



https://huggingface.co/spaces/open-llm-leaderboard/blog

4. Separating words

Separating words: what for and why is it hard?

Needed for

- Heuristic filters
- Deduplication
- Evaluations

Challenges:

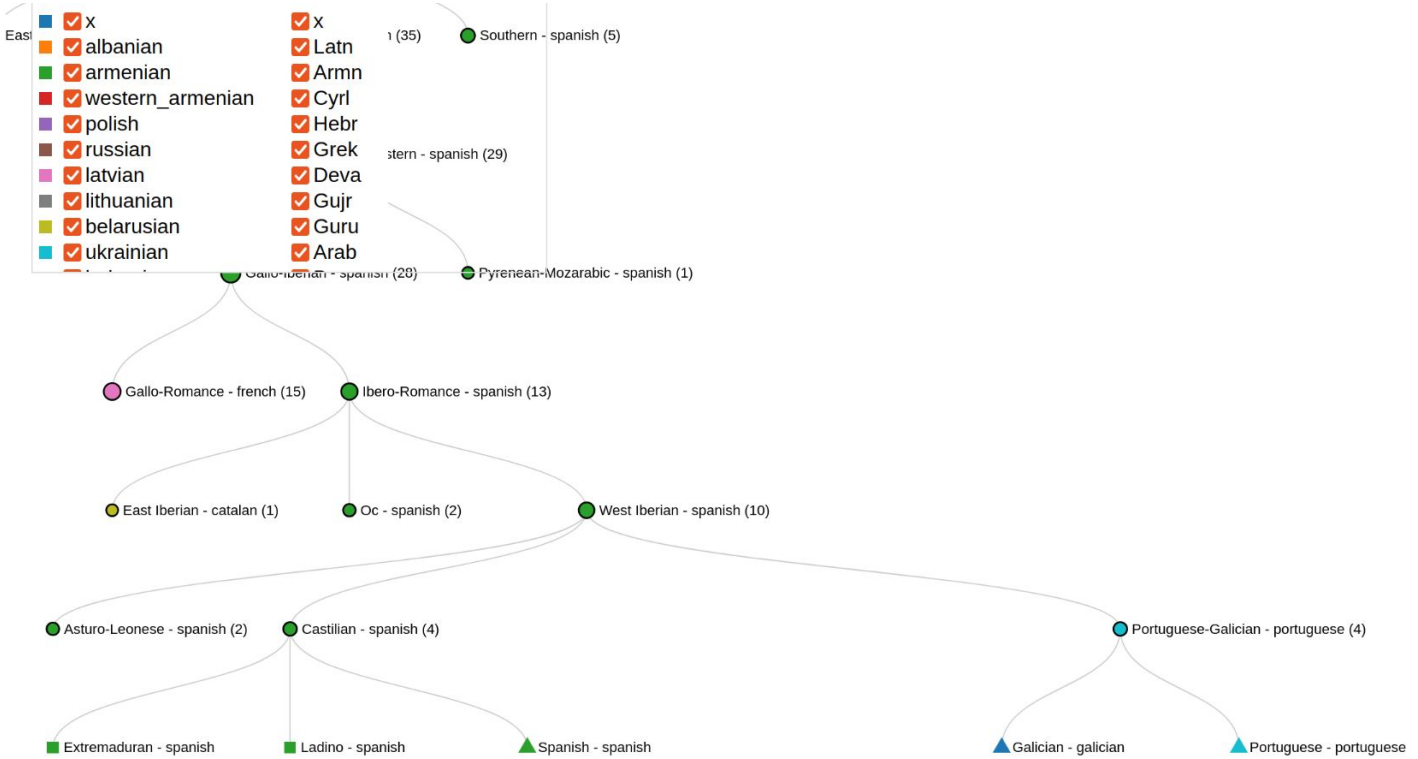
- Many writing systems have diff word boundary chars (Ethiopic)
- Some have no boundaries (Chinese, Japanese, Korean)
- Even for English, “space + punct” not enough

Use word tokenizers/segmentors

Native word tokenizers

- SpaCy
- Stanza
- IndicNLP (indic languages)
- PyThaiNLP (Thai)
- Kiwipiepy (Korean)
- KhmerNLTK (Khmer)
- Botok (Tibetan)
- …

Proxy word tokenizers



https://www.ethnologue.com/browse/families/

Tokenizer assignments

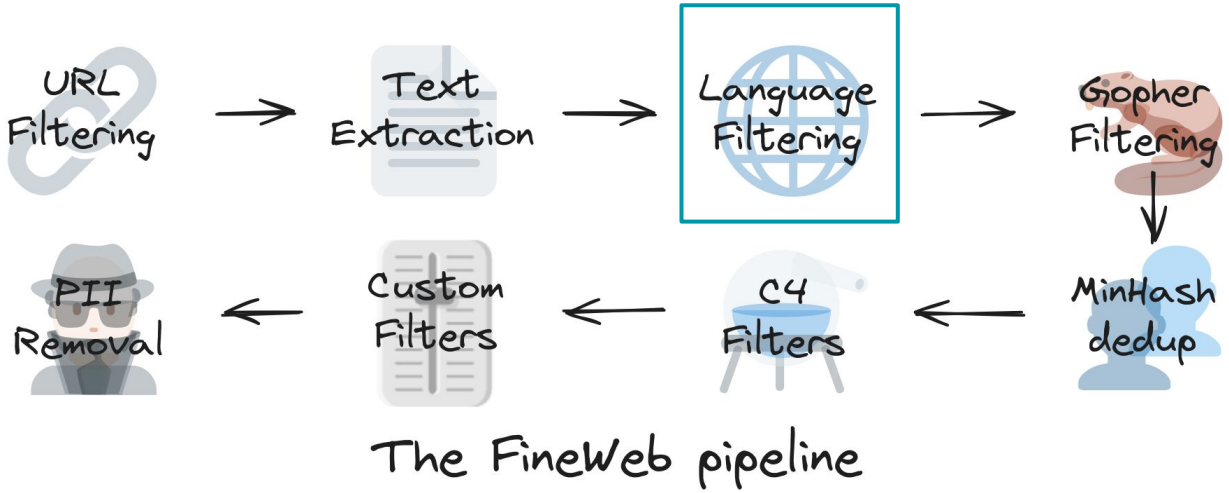
1. Assign native
2. Family tree method
3. Biggest language per script

bco		Latn	Kaluli	Trans-New Guinea	SpaCyTokenizer	xx
bcw		Latn	Bana	Afro-Asiatic	SpaCyTokenizer	xx
bdd		Latn	Bunama	Austronesian	SpaCyTokenizer	ms
bdh		Latn	Baka (South Sudan)	Nilo-Saharan	SpaCyTokenizer	xx
bdq		Latn	Bahnar	Austro-Asiatic	SpaCyTokenizer	vi
bea		Latn	Beaver	Eyak-Athabaskan	SpaCyTokenizer	xx
bef		Latn	Benabena	Trans-New Guinea	SpaCyTokenizer	xx
bel	be	Cyrl	Belarusian	Indo-European	StanzaTokenizer	be
bem		Latn	Bemba (Zambia)	Niger-Congo	SpaCyTokenizer	tn
ben	bn	Beng	Bengali	Indo-European	IndicNLPTokenizer	bn
ben	bn	Latn	Bengali	Indo-European	StanzaTokenizer	kmr
beq		Latn	Beembe	Niger-Congo	SpaCyTokenizer	tn
bew		Latn	Betawi	Creole	SpaCyTokenizer	ms
bex		Latn	Jur Modo	Nilo-Saharan	SpaCyTokenizer	xx
bfd		Latn	Bafut	Niger-Congo	SpaCyTokenizer	tn
bfo		Latn	Malba Birifor	Niger-Congo	SpaCyTokenizer	tn
bgr		Latn	Bawm Chin	Sino-Tibetan	SpaCyTokenizer	xx
bgs		Latn	Tagabawa	Austronesian	SpaCyTokenizer	tl
bgt		Latn	Bughotu	Austronesian	SpaCyTokenizer	ms

Short questions break

5. Building 

Starting point: FineWeb



The FineWeb pipeline

~60% extracted text from ~100 CommonCrawl snapshots

***Approach**: adapt the pipeline as closely as possible*

5.1 Language Identification

Language Identification (LID)

- Limits languages that we can process
- Affects quality of predictions

Classifiers:

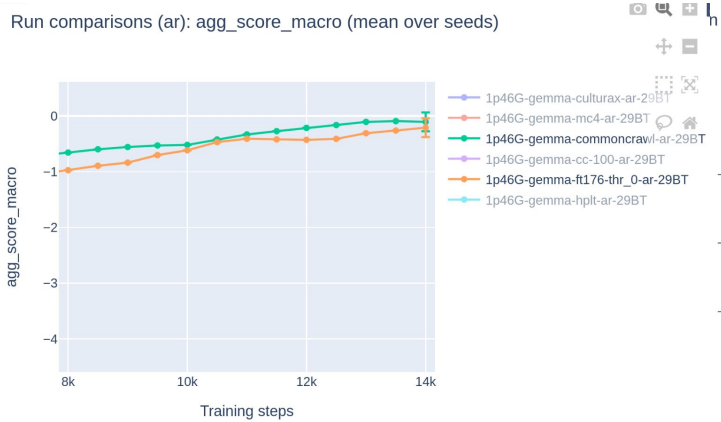
- CLD2/3 (106 languages)
- fastText based
  - FT176 (CC-Net) (176)
  - OpenLID (193)
  - GlotLID (1880)
- Transformer based (*expensive and slow*)

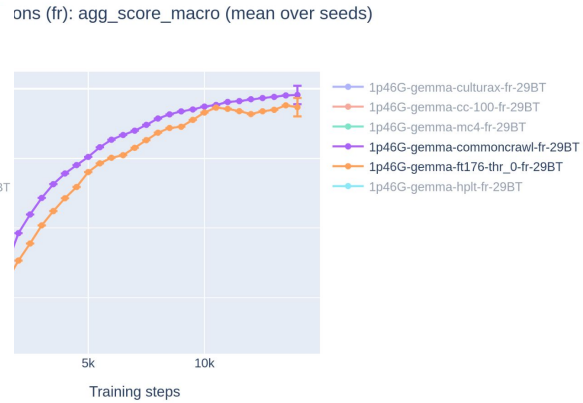
The case for GlotLID

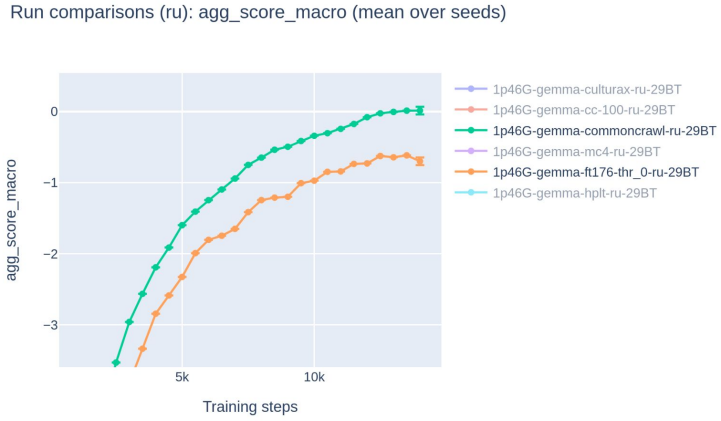
- Allows us to support many more languages (including low-res)
- Reduce “out-of-model cousin” errors [Caswell et al. 2020, Kreutzer et al. 2022]
- Script detection
- Labels contain script
- UND label
- UNK/noise labels
- They report strong benchmarks

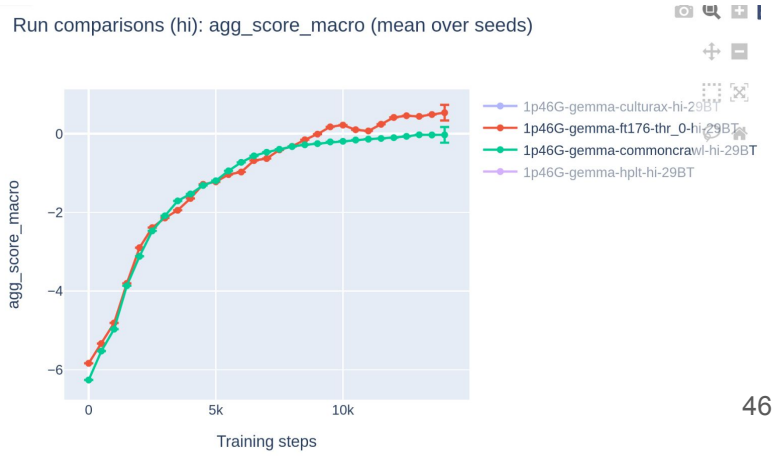
What about model performance?

GlotLID vs fastText









LID confidence thresholds

- Most works use a single threshold for all languages (~0.5)
- Not all languages are equal

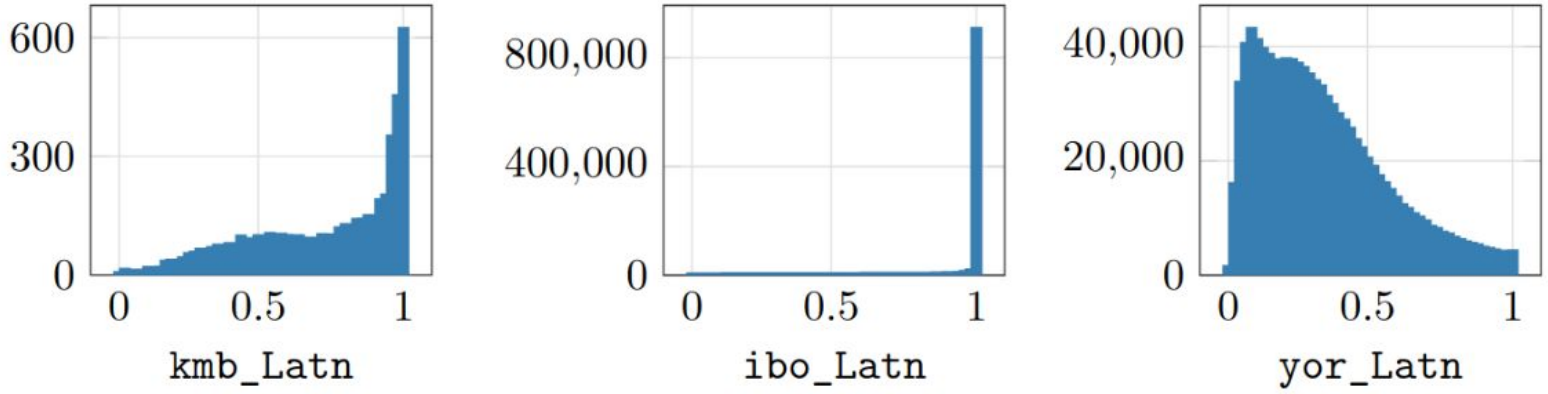
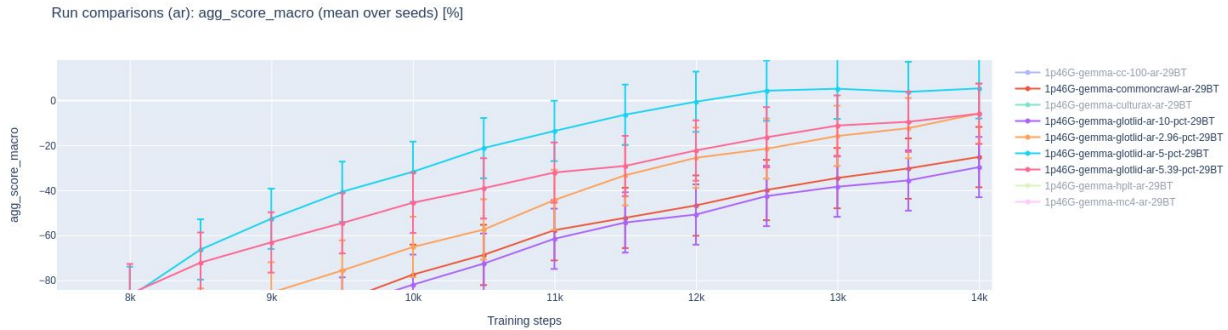


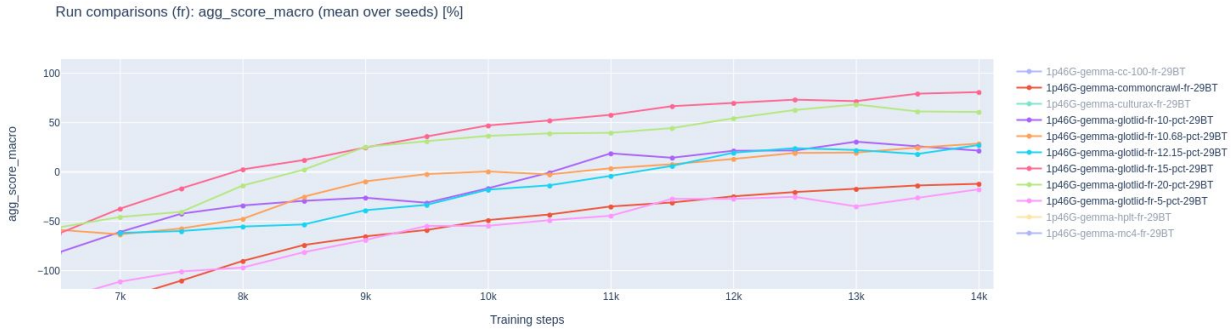
Figure 11: **LID Score Distribution Patterns on ParaCrawl**, illustrated with Kimbundu, Igbo and Yoruba.

Costa-jussà et al. (2022) No Language Left Behind: Scaling Human-Centered Machine Translation

Testing different thresholds/removal %s

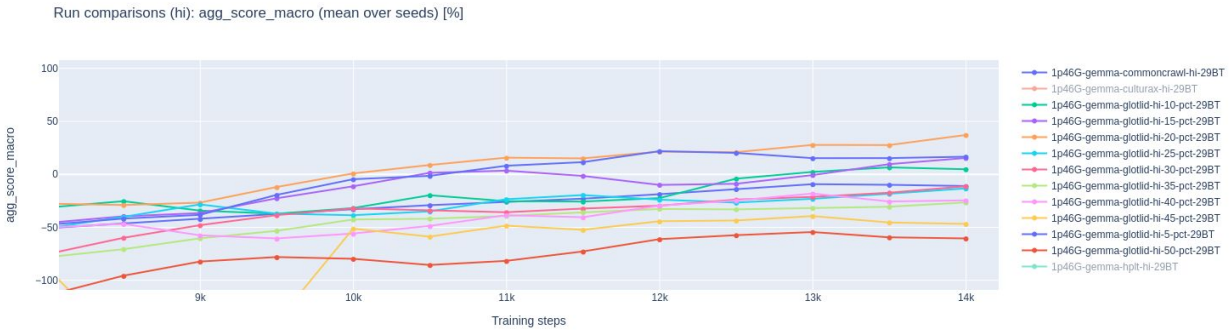


5% ~ 0.9



15% ~ 0.87

Testing different thresholds/removal %s



20% ~ 0.65

64.24% ~ 0.3 (1.6 epochs)

Thresholds summary

Language	Min	Max	Chosen	Formula
Arabic	0.7	0.9	0.9	0.8812
Chinese	0.9	0.94	0.9	0.7415
French	0.8	0.93	0.85	0.8195
Hindi	0.6	0.7	0.65	0.6827
Russian	-	-	0.9	0.9
Swahili	0.186	0.544	0.3	0.3
Telugu	0.7	0.996	0.7	0.7002
Thai	0.85	0.961	0.9	0.9
Turkish	0.75	0.85	0.8	0.8753

Formula: *min(max(median - std_dev, 0.3), 0.9)*

Thresholds summary

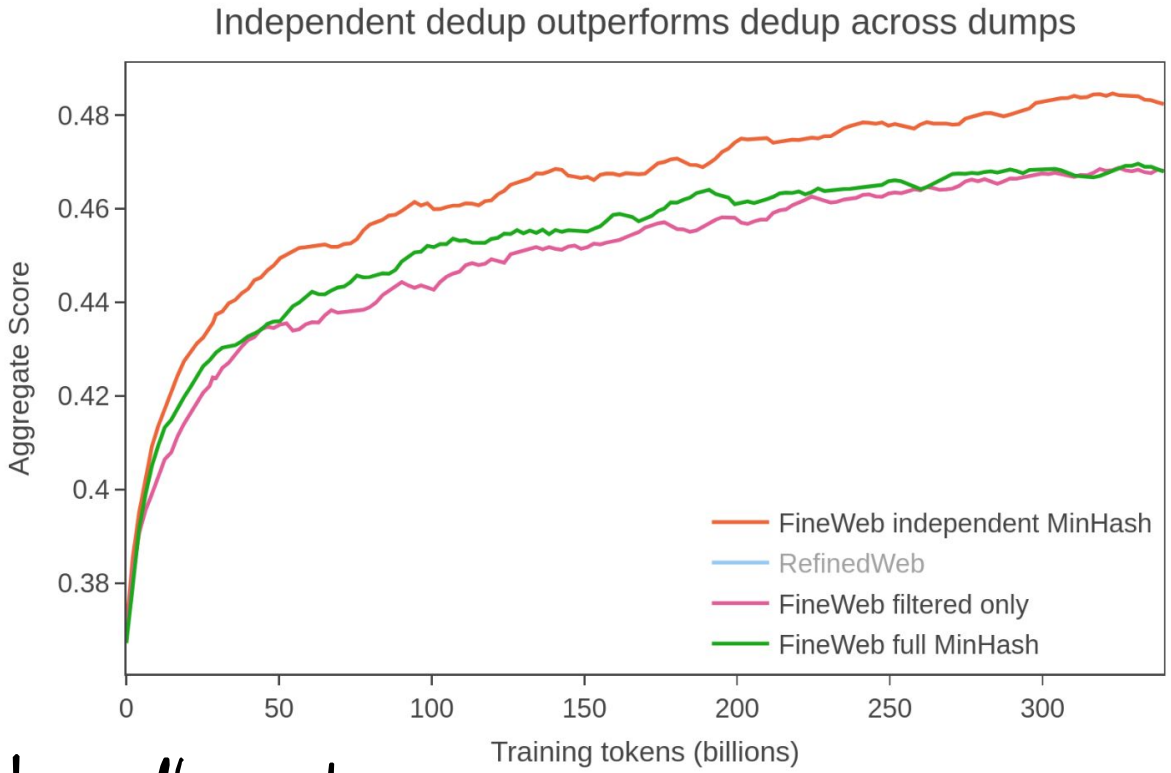
code	script	mean	std	min	q1	median	q3	max	selection	pct	size_left
rus	Cyrl	0.9796	0.0885	0.0130	0.9970	0.9990	1.0000	1.0000	**0.9000**	4.52%	15.65TB
jpn	Jpan	0.9731	0.1136	0.0200	0.9990	1.0000	1.0000	1.0000	**0.8864**	5.60%	7.77TB
cmn	Hani	0.8212	0.2523	0.0100	0.7800	0.9440	0.9850	1.0000	**0.6917**	20.04%	7.70TB
deu	Latn	0.9263	0.1739	0.0230	0.9750	0.9950	0.9980	1.0000	**0.8211**	12.68%	7.45TB
spa	Latn	0.9400	0.1546	0.0260	0.9830	0.9950	0.9970	1.0000	**0.8404**	10.59%	6.82TB
fra	Latn	0.9258	0.1715	0.0200	0.9660	0.9960	0.9980	1.0000	**0.8245**	12.71%	6.02TB
por	Latn	0.9323	0.1987	0.0200	0.9940	0.9980	0.9990	1.0000	**0.7993**	9.48%	3.22TB
ita	Latn	0.9187	0.1734	0.0220	0.9550	0.9900	0.9950	1.0000	**0.8166**	13.14%	3.22TB
pol	Latn	0.8537	0.3107	0.0170	0.9860	1.0000	1.0000	1.0000	**0.6893**	17.07%	2.42TB
nld	Latn	0.8393	0.3078	0.0090	0.9400	0.9970	0.9990	1.0000	**0.6892**	19.35%	2.25TB
ind	Latn	0.8175	0.2644	0.0210	0.8020	0.9490	0.9770	1.0000	**0.6846**	19.04%	1.50TB
tur	Latn	0.9554	0.1243	0.0140	0.9900	0.9990	1.0000	1.0000	**0.8747**	10.44%	1.41TB
kal	Latn	0.2501	0.3093	0.0080	0.0640	0.1080	0.2330	1.0000	**0.3000**	78.90%	1.88GB
acm	Arab	0.5124	0.1841	0.0180	0.3720	0.4880	0.6260	1.0000	**0.3039**	9.72%	1.72GB
spa	Latn	0.3352	0.3418	0.0110	0.1080	0.1660	0.4260	1.0000	**0.3000**	69.46%	1.71GB

5.2 Deduplication

PARAMS : n-gram size
: hashes

Deduplication

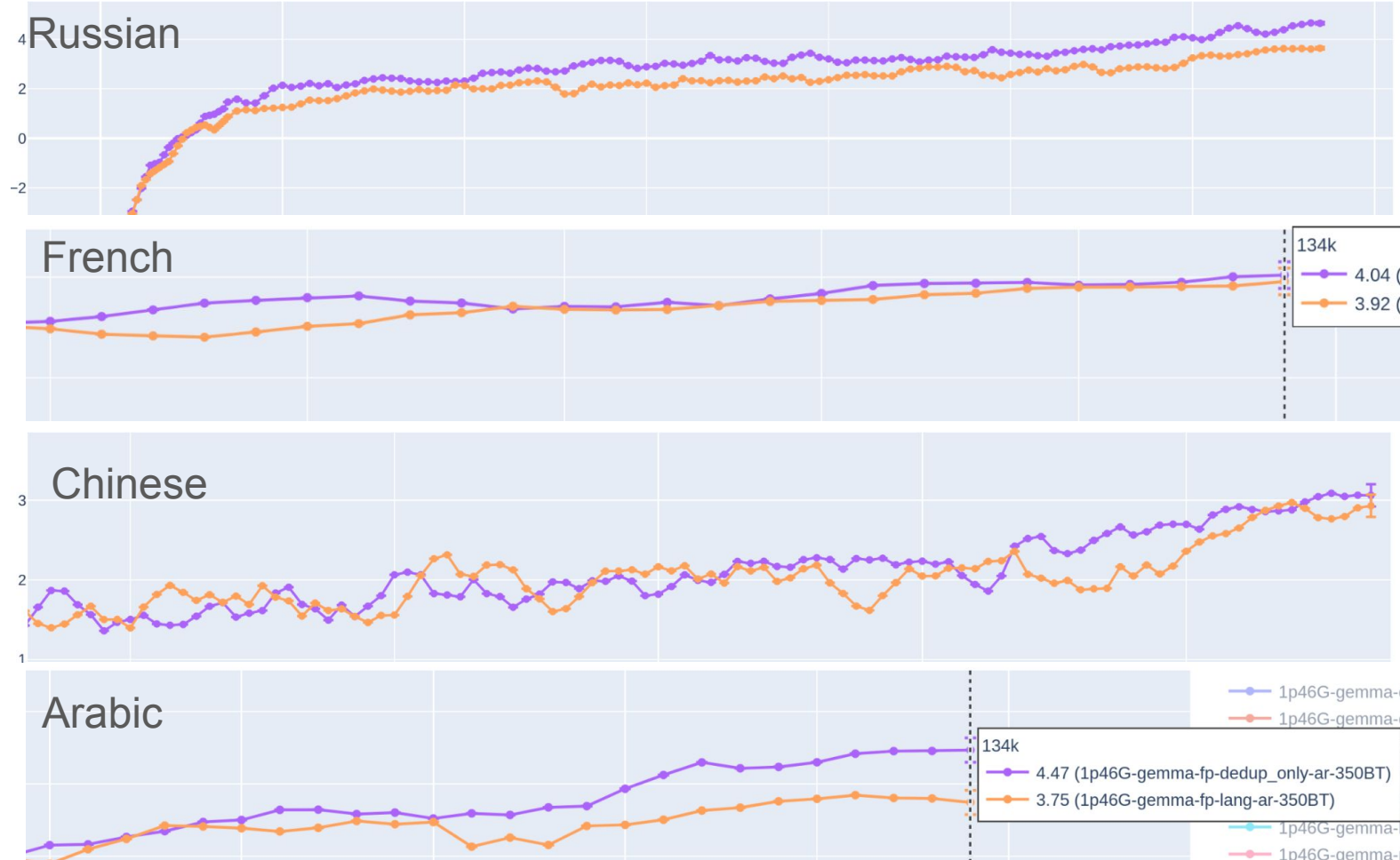
- Per language
- Global vs per shard questions from FineWeb
- Free upsampling/quality signal?



Compromise: save **cluster size**

→ MinHash allows this

Different impact per language



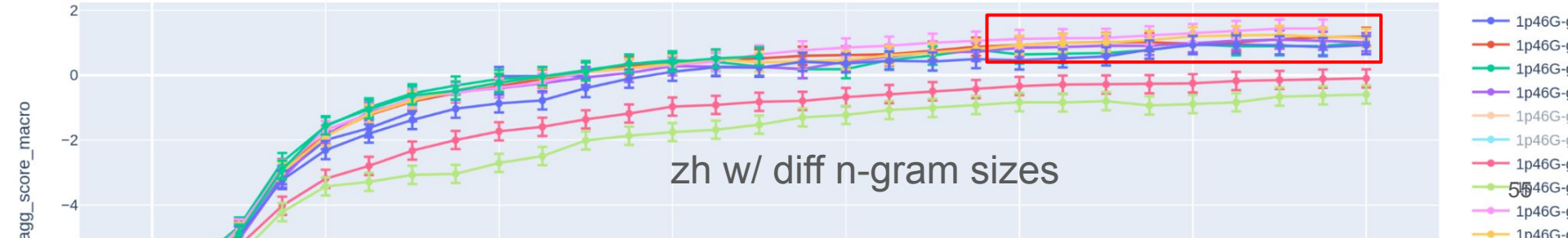
Different impact per language

- Big boost: russian and turkish
- Small boost: arabic and french
- No change: chinese and thai

- Low vs high res?
- Difference based on word sizes?
  - 3-10 ngram size for zh: no change (2%)

Avg word length

```
cmn_Hani, 2.071644
tha_Thai, 4.418740
arb_Arab, 4.506921
#hin_Deva, 4.933885
eng_Latn, 5.009773 # for reference
fra_Latn, 5.043927
#swh_Latn, 5.120833
rus_Cyrl, 6.158980
tur_Latn, 6.588550
#tel_Telu, 6.928036
```



5.3 Heuristic filters

Heuristic filters

Heuristic Category	Common Utility Functions	Example Selection Mechanisms
Item Count	# of characters in a {word/line/paragraph/document} # of {words/lines/sentences} in a document	Remove documents with fewer than 5 words (Raffel et al., 2020)
Repetition Count	# of times a {character/n-gram/word/ sentence/paragraph} is repeated	Remove lines that repeat the same word more than 4 times consecutively (Laurençon et al., 2022)
Existence	Whether a {word/n-gram} is in the document Whether a terminal punctuation is at the end of a line	Remove lines starting with "sign-in" (Penedo et al., 2023)
Ratio	% of alphabetic characters in a document % of numerals/uppercase characters in a {line/document}	Remove documents with a symbol-to-word ratio greater than 0.1 (for "#" and "...") (Rae et al., 2021)
Statistics	The mean length (and standard deviation) of all lines in a document	Remove code files that have mean line length greater than 100 characters (Chen et al., 2021)

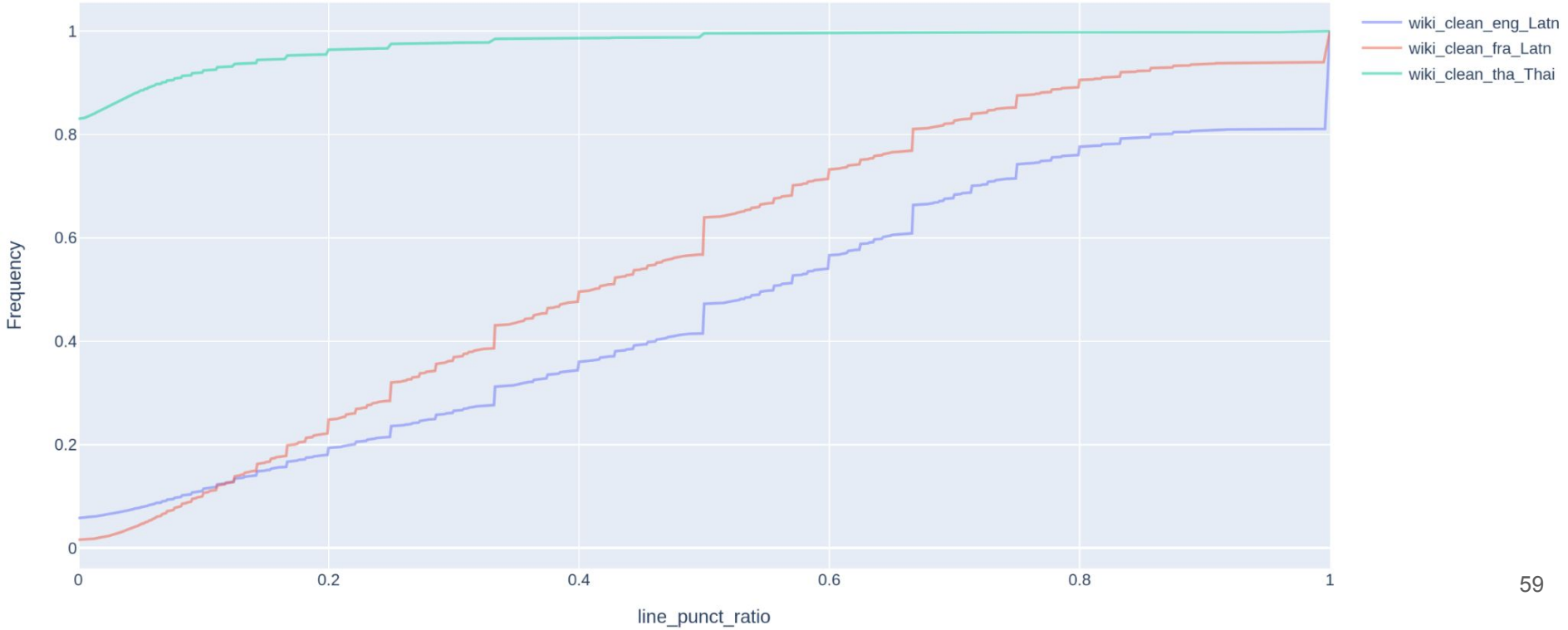
Table 2: **Commonly used heuristic utility functions and demonstrative selection mechanisms.**

Albalak et al. (2024) "A Survey on Data Selection for Language Models"

Filtering: adapting filters

- Word tokenizers
- Character sets (terminal punctuation, etc)
- Stopwords
- Adapting thresholds
  - Ground truth/reference data?
  - Formulas/approach?
- Filters to adapt:
  - Gopher quality
  - Gopher repetition
  - C4 filters
  - FineWeb filters
- Loooots of experiments

Adapting based on distributions



Adaptation recipes

- Quantiles
- Mean ± n*std_dev
- Median ± n*std_dev
- 10% [CulturaX, Nguyen et al. (2023)]
- Median ratio [HPLT2, Gibert et al. (2024)*]

- Wikipedia
- Actual CommonCrawl data
- GlotLID training corpus

Filtering results

- Diff methods better for diff things
- Some better on web vs wiki

Final decisions:

- Drop C4 filters
- 10% for repetition on web
- Quantile with wiki on the rest

Filter: fwq

	method	score	avg_rank	avg_per_lang_rank
0	wiki-mstd	0.645576	1	3.777778
1	wiki-q	0.631513	2	3.333333
2	ccg-q	0.376301	3	7.555556
3	ccg-mdmad	0.348225	4	6.777778
4	wiki-mdstd	0.313775	5	7.444444
5	ccg-mmad	0.290234	6	8.111111

Filter: goq

	method	score	avg_rank	avg_per_lang_rank
0	wiki-q	0.612105	1	5.111111
1	ccg-q	0.401157	2	8.444444
2	wiki-mdstd	0.333273	3	7.777778
3	ccg-mdmad	0.214111	4	9.333333
4	wiki-mstd	0.195092	5	9.333333
5	ccg-mmad	0.168075	6	9.777778

Stopwords

- Extra LID fixer
- Very sensitive to corpus contamination
- Issues with common words across languages

	Code	Script	Name	Original Docs	Original Disk size	Goq docs	Goq doc -%	Goq doc len	Goq doc len -%
2	~~eng~~	~~Latn~~	~~English~~	~~21719803927~~	~~21.24TB~~		-	-	-
3	rus	Cyrl	**Russian**	11872796092	16.39TB	573483034	0.1665445467	2799230670509	**0.1135141655**
4	cmn	Hani	**Mandarin Chinese**	6327989235	9.63TB	534797244	13.27%	785415438595	**10.68%**
5	deu	Latn	German	8735981870	8.53TB	400170788	21.16%	1310701867405	**11.35%**
6	jpn	Jpan	Japanese	7078824012	8.24TB	322597400	20.53%	437094118097	**17.82%**
7	spa	Latn	Spanish	7543940625	7.63TB	387929873	0.2199079708	1208961845618	**0.1660536802**
8	fra	Latn	**French**	7043433981	6.90TB	313554322	27.35%	1019129012839	**17.33%**
9	ita	Latn	Italian	3727916440	3.70TB	208611881	0.2199161279	665895543015	**0.1383525421**
10	por	Latn	Portuguese	3706687076	3.56TB	179744148	0.3061009984	513161069768	**0.2244903283**
11	pol	Latn	Polish	2795766947	2.92TB	129198073	24.01%	361985297759	**18.43%**
12	nld	Latn	Dutch	3275490629	2.79TB	127310120	26.09%	360329043265	**15.24%**
13	dag	Latn	Dagbani	1752265031	2.24TB	27709	**99.35%**	173967251	**97.30%**

5.4 Rehydration: free boost

Back to dedup

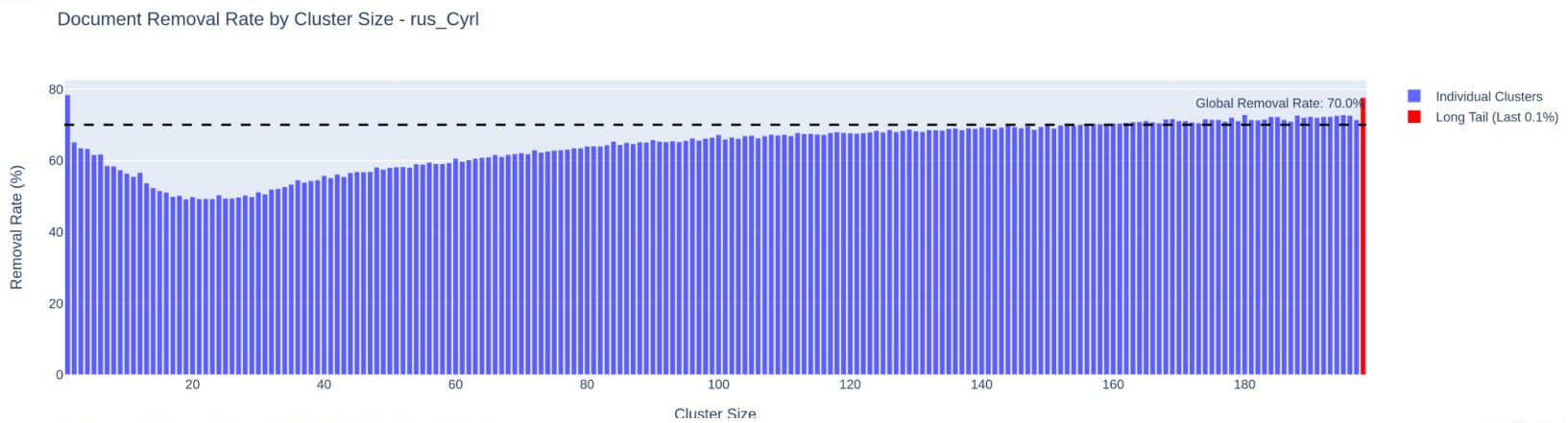
Remember cluster sizes? What if we “rehydrate” the dataset?

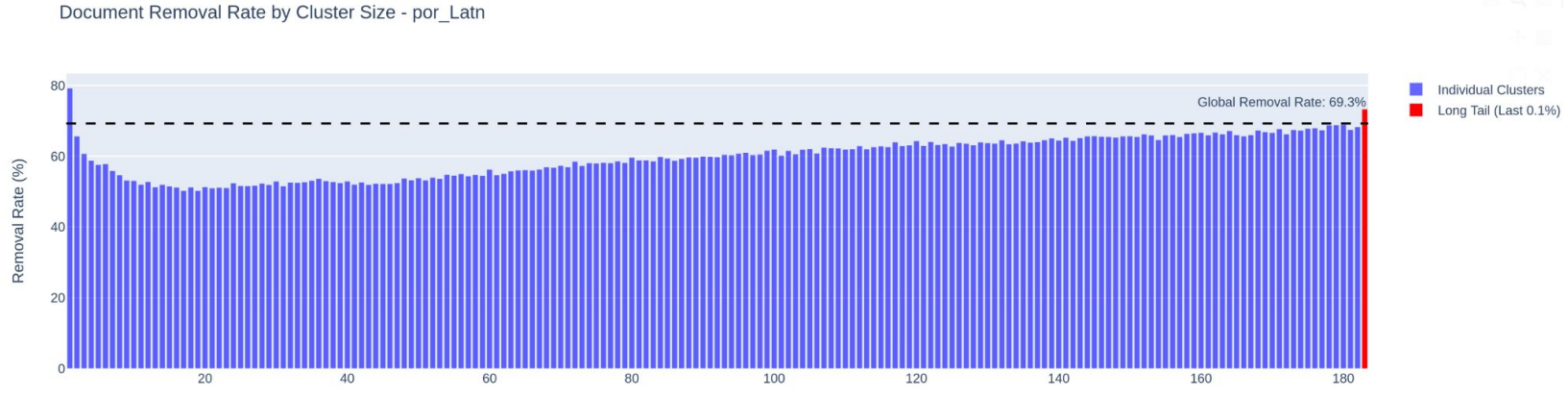
Motivated by the FineWeb study, we opted to upsample documents based on their natural distribution. However, since duplication is only an indirect indicator of quality, we upsample documents to a few predefined levels rather than using their exact count. Specifically, we set the upsampling weight to 3 for documents with 2 to 5 duplicates, 5 for those with 5 to 100 duplicates, 8 for 101 to 1000 duplicates, and 10 for documents with over 1000 duplicates. These values were selected heuristically and informed by preliminary small-scale experiments. For non-CommonCrawl data sources, we assign a weight of 2 if the document appears more than once. This straightforward approach results in a corpus exceeding 15 trillion tokens, making it one of the largest open-access pre-training datasets available.

https://huggingface.co/spaces/LLM360/TxT360

Quality by cluster size

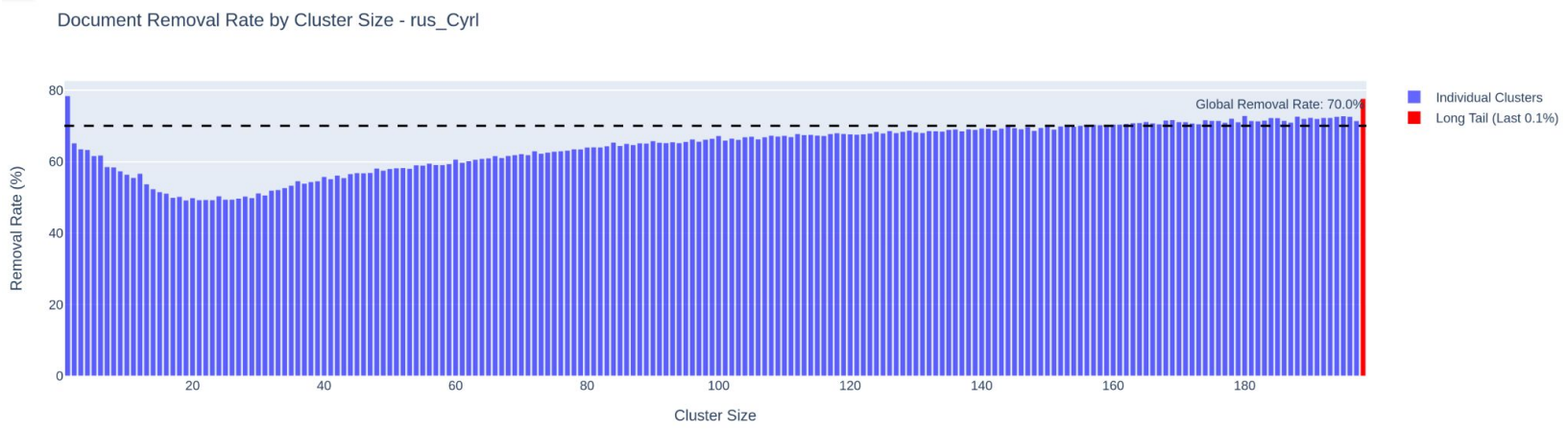
Matches experimental results!

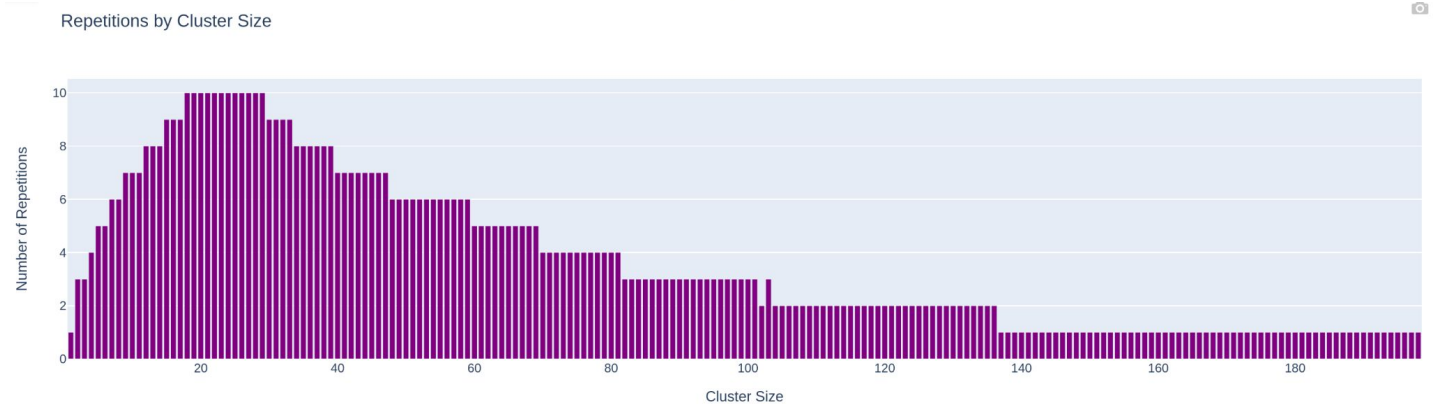




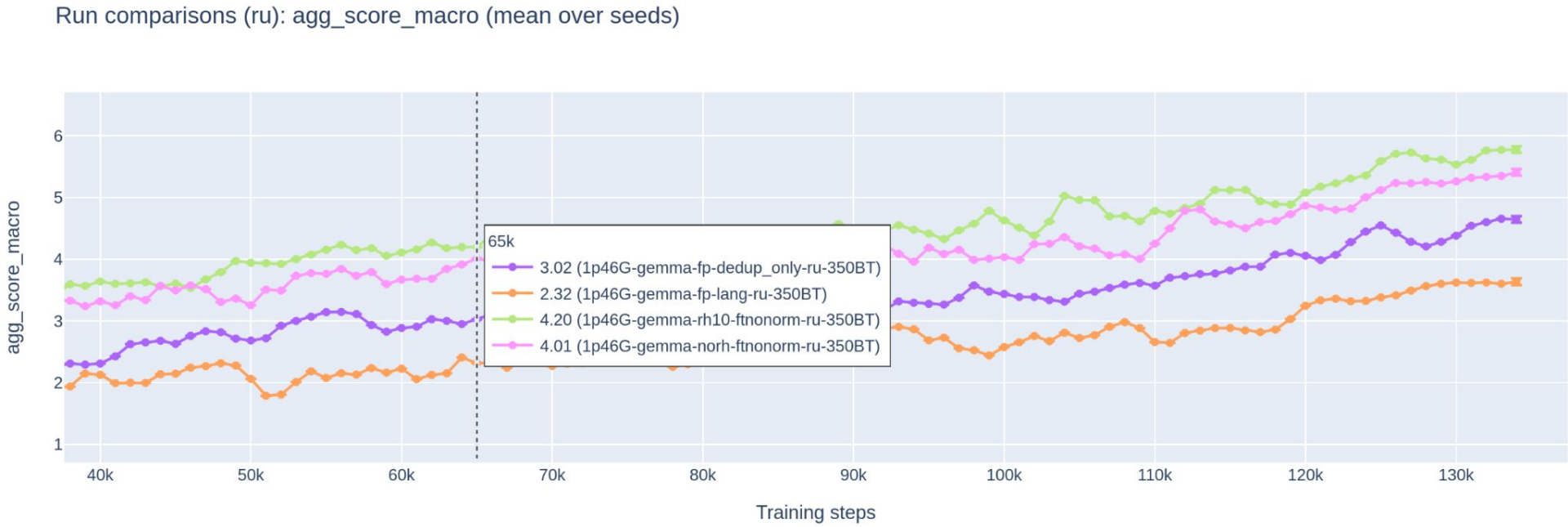
Quality by cluster size

Matches experimental results!



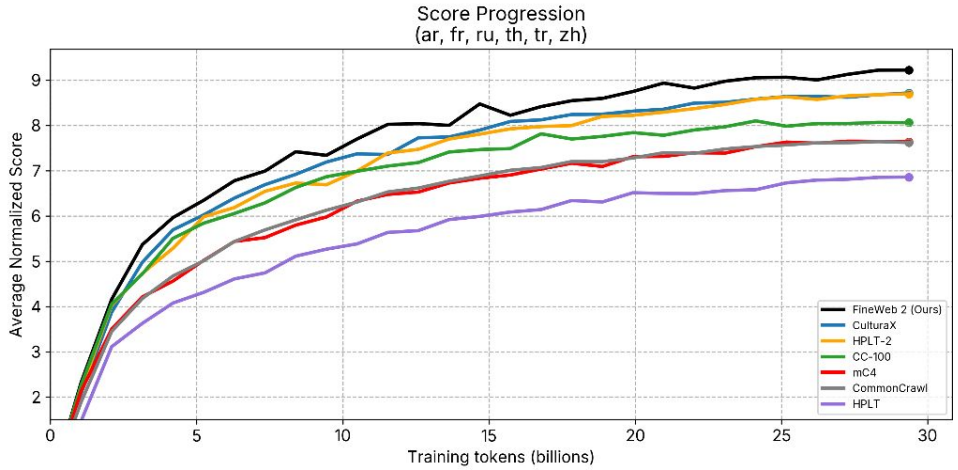


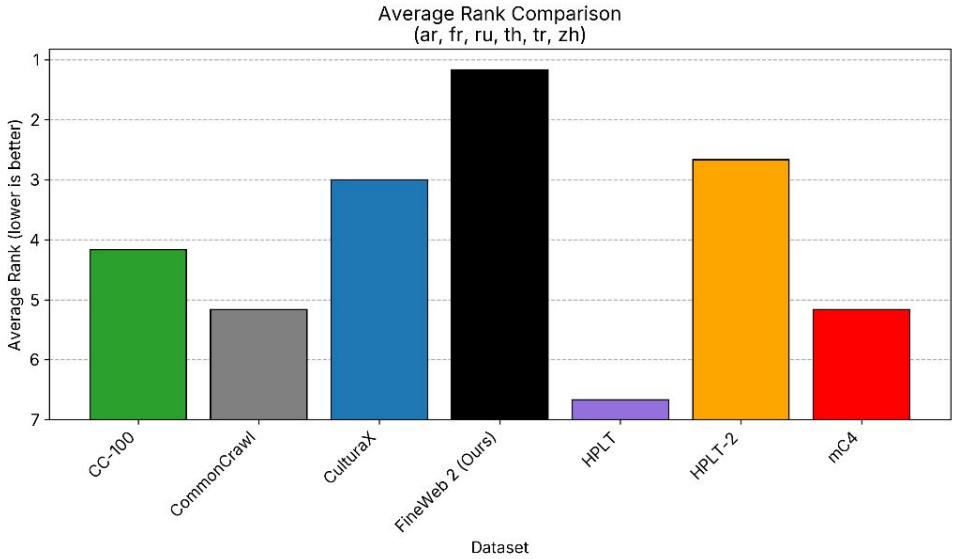
Iterative improvements



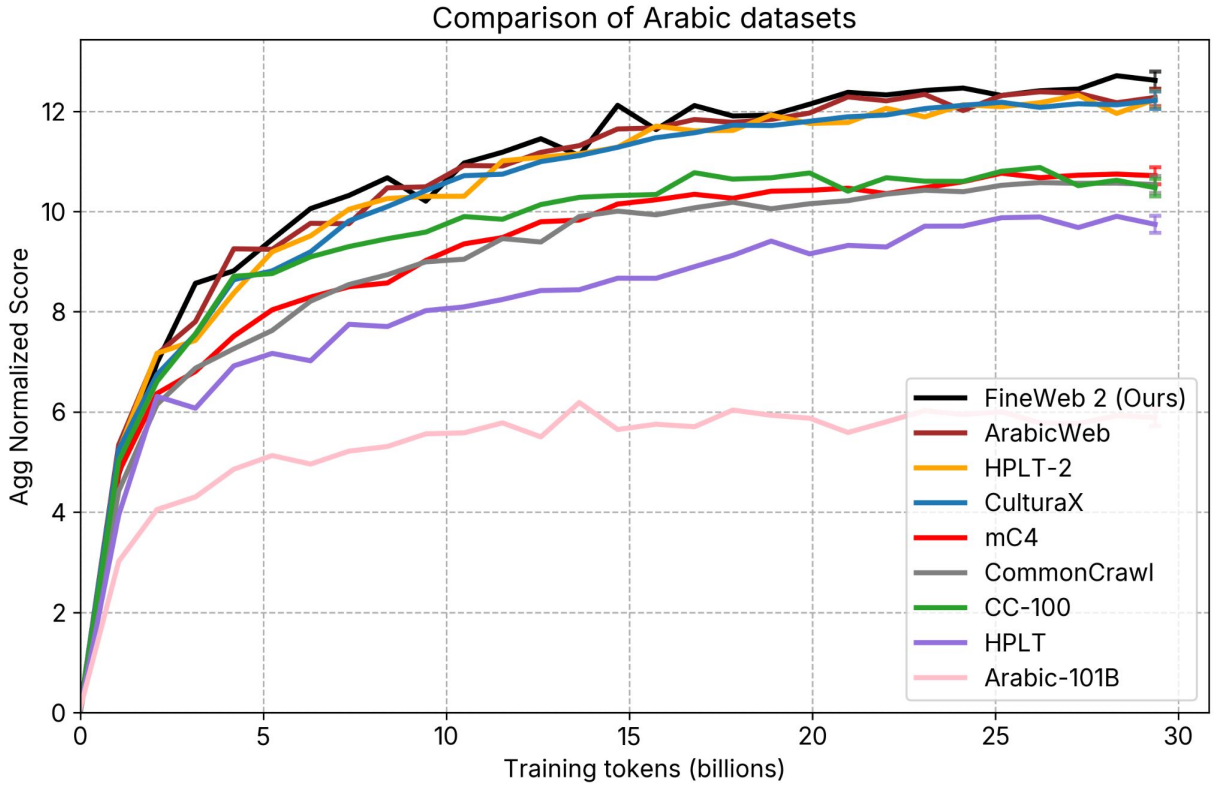
6. Conclusion

Final comparisons

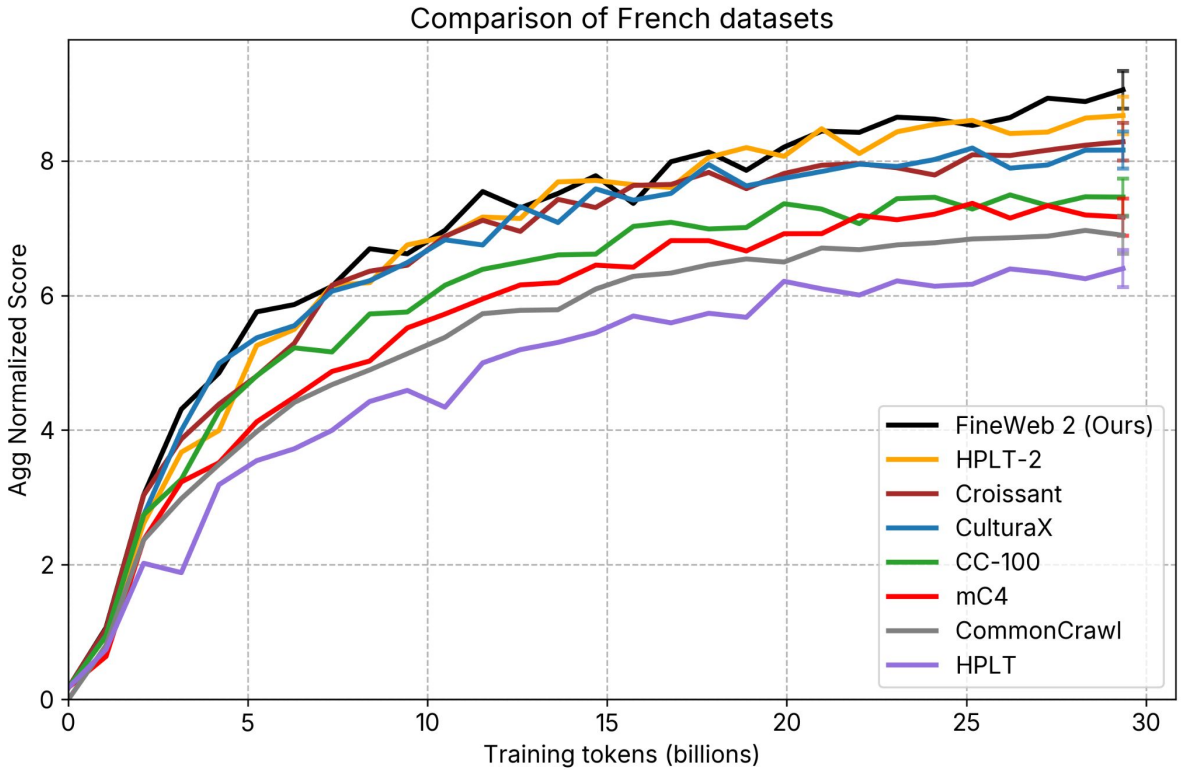




Comparisons (ar)

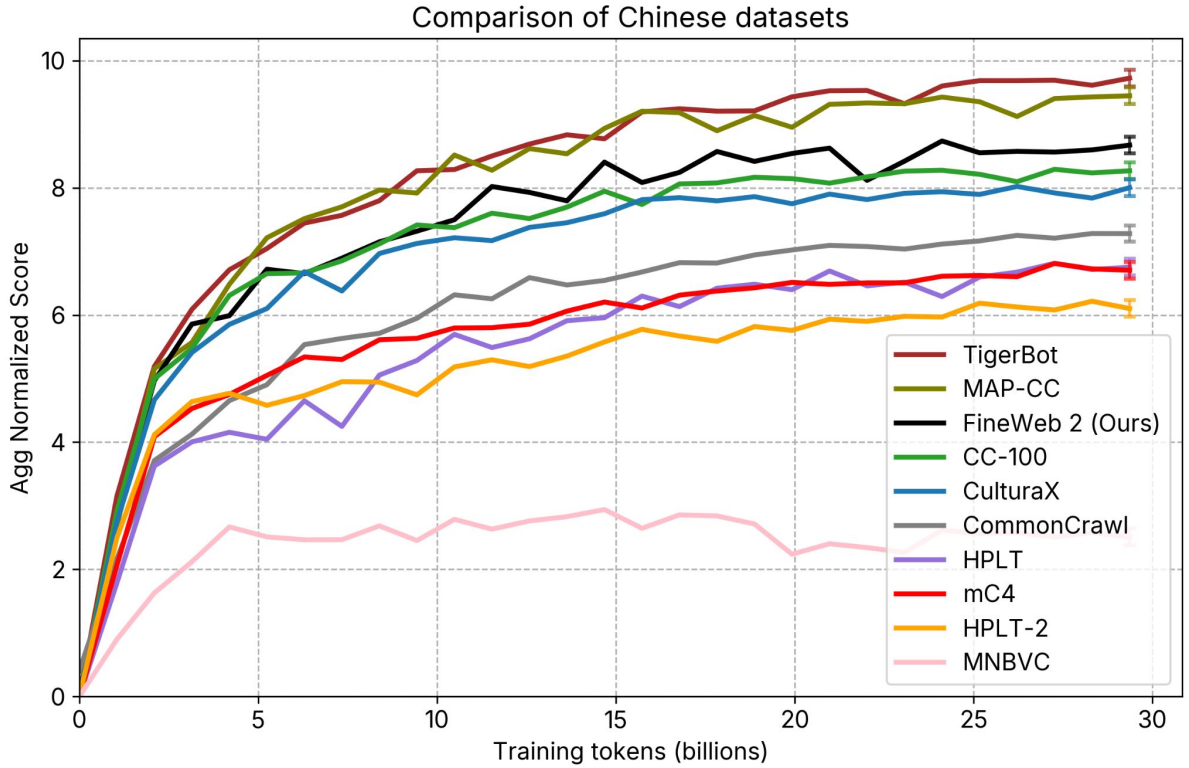


Comparisons (fr)



Comparisons (zh)

? ! :
vs
? ! :



Fully reproducible

```
"""
    2. We then applied GlotLID (we actually applied it to all dumps)
"""
GLOTLID_OUTPUT_PATH = f"{MAIN_OUTPUT_PATH}/glotlid"
for dump in [
    "CC-MAIN-2023-50",
    # ...
]:
    SlurmPipelineExecutor(
        job_name=f"glotlid_{dump}",
        pipeline=[
            JsonlReader(f"{BASE_OUTPUT_PATH}/2_non_english/{DUMP_TO_PROCESS}"),
            # we keep annotations of alternative labels that are classified above 0.01
            # backend glotlid instead of ft176
            LanguageFilter(backend="glotlid", label_only=True, keep_top_pairs_threshold=0.01),
            # save in "language_script/dump"
            JsonlWriter(GLOTLID_OUTPUT_PATH,
                        output_filename="${language}_${language_script}/" + dump + "/${rank}.jsonl.gz")
        ],
        tasks=1000,
        # workers=50,
        mem_per_cpu_gb=4,
        logging_dir=f"{LOGS_PATH}/glotlid/{dump}",
        partition="hopper-cpu",
        randomize_start_duration=5 * 60,
        time="10:00:00",
    ).run()

"""
    From this point on, processing is PER LANGUAGE
"""
```

main ▾ **fineweb-2** / configs / **por_Latn.yml**

Blame 44 lines (44 loc) · 435 Bytes

```
  - - 9
    - 0.131
  - - 10
    - 0.12
language_score: 0.799
line_punct_thr: 0.154
max_avg_word_length: 13
max_non_alpha_words_ratio: 0.814
min_avg_word_length: 3
new_line_ratio: 0.23
stopwords:
- de
- a
- e
- o
- em
- do
- da
- que
- um
- 'no'
- uma
- com
- para
- na
- "\xE9"
- foi
```

The end

Dataset

https://huggingface.co/datasets/HuggingFaceFW/fineweb-2

Code/configs
https://github.com/huggingface/fineweb-2

FineWeb v1 blogpost

https://huggingface.co/spaces/HuggingFaceFW/blogpost-fineweb-v1

FineWeb2 blogpost: coming soon