HAIDASS: BILINGUAL SMALL LANGUAGE MODELS BASED ON DATA CURATION AND LANGUAGE SCHEDULING

Anonymous authors

ABSTRACT

Small language models (SLMs) offer clear advantages in efficiency and deployment, especially for on-device bilingual applications, but at the sub-200M scale, they remain English-centric: robust bilingual SLMs are scarce. Introducing a second language at this scale forces both languages to share a narrow parameter budget. To inform the design of such a bilingual SLM, we study how Chinese-English pretraining corpora interact. We quantify the cross-lingual interference between the two languages and how it depends on the mixing ratio, and compare several data strategies. Guided by this, we curate a balanced Chinese-English corpus and train Haidass-143M on 391.1B tokens, the first bilingual SLM at this scale, with a multi-stage real-to-synthetic data curation and language scheduling. It ranks 5th on the Open SLM Leaderboard. Building on this model, we post-train Haidass-Translate-143M. With direction-specific beam decoding, it achieves 30.36 BLEU and 21.40 chrF++ on FLORES-200 (en→zh), outperforming the dedicated NLLB-200-distilled-600M by 7.92 BLEU and 4.66 chrF++, and essentially matches Qwen3-0.6B (30.94 BLEU, 21.10 chrF++) despite roughly 4× fewer parameters. This work provides empirical guidance for the training recipe design of SLMs.

1 INTRODUCTION

Modern large language models (LLMs), from GPT (Achiam et al., 2023) to Qwen (Yang et al., 2025) and DeepSeek (Guo et al., 2025), have scaled to billions or trillions of parameters and achieve state-of-the-art results across general language tasks. However, their parameter and compute demands make them costly to serve and impractical on edge terminals, localized servers, and low-resource deployments.

This gap has motivated sub-billion small language models (SLMs) such as Gemma-3 (Kamath et al., 2025), SmolLM2 (Ben Allal et al., 2025), and MobileLLM-R1 (Zhao et al., 2026), which trade capacity for low latency and small footprints. These works have advanced SLMs through architectural design, scalable training and data strategies (Liu et al., 2024; Hu et al., 2024; Ben Allal et al., 2025). However, these advances have largely been demonstrated in monolingual or English-centric settings, leaving bilingual modeling underexplored.

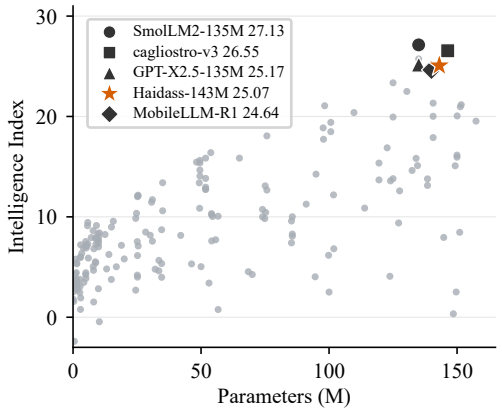

Figure 1: Haidass-143M in the Open SLM Leaderboard. The legend lists selected models and their Intelligence Index.

Within a limited parameter budget, bilingual pretraining requires consideration of cross-lingual interactions and training data. Prior work has shown that multilingual training can produce either positive transfer and interference, depending on

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

data availability and mixture, linguistic similarity and model scale (Chang et al., 2024; Liu et al., 2020; Conneau et al., 2020). Data curation and language scheduling are two relevant directions for improving SLM performance. Corpus filtering can improve pretraining performance (Li et al., 2024b; Penedo et al., 2024), while synthetic data can supplement web text with targeted content and alternative forms of supervision.Multi-stage training allows these data sources to be integrated at different stages, as explored in SmolLM2 (Ben Allal et al., 2025). In bilingual training, language ordering is also critical, given the difficulty of retaining previously acquired knowledge under sequential exposure to different languages (M'hamdi et al., 2023; Winata et al., 2023). These considerations raise two questions: (1) how do data curation and language scheduling affect bilingual pretraining, and (2) how should filtered natural language text and synthetic data be integrated into bilingual SLM training?

To address these questions, we investigate cross-lingual interactions in SLMs across multiple scales through controlled comparisons of data curation and language scheduling. We find that higher-quality data improves overall performance across model scales, although its benefits vary with model capacity and target capability, while continuous language interleaving preserves bilingual competence more effectively than blockwise or sequential training. Guided by these findings, we develop a natural-to-synthetic bilingual training recipe that maintains exposure to both languages while progressively introducing more structured and synthetic data. This training recipe provides a practical framework for jointly balancing data quality, capability coverage, and bilingual retention in SLM pretraining.

Finally, we present Haidass, a fully open-source Chinese-English bilingual SLM family trained on Ascend 910B. The family includes the base model Haidass-143M and the translation model Haidass-Translate-143M. Haidass-143M is trained with our Natural-to-Synthetic Data Curation and Language Scheduling, which integrates filtered natural language text and synthetic data under a fixed token budget. It ranks 5th on the Open SLM Leaderboard (25.07), outperforming MobileLLM-R1. With direction-specific beam decoding, Haidass-Translate-143M achieves 30.36 BLEU and 21.40 chrF++ (en→zh), and 20.62 BLEU and 46.51 chrF++ (zh→en) on FLORES-200. Its en→zh result exceeds NLLB-200-distilled-600M by 7.92 BLEU, while its bidirectional BLEU essentially matches Qwen3-0.6B despite roughly $4\times$ fewer parameters.

Our contributions are as follows:

- We study cross-lingual interactions in sub-200M-scale SLMs, specifically the trade-offs between Chinese and English language modeling performance and the impact of language scheduling on bilingual retention.
- We develop a Natural-to-Synthetic data curation for bilingual pretraining, progressing from predominantly filtered natural language text to mixtures with greater emphasis on synthetic rewritten and question–answer data.
- We release Haidass, a Chinese–English bilingual SLM family trained on Ascend 910B. Haidass-143M ranks 5th on the Open SLM Leaderboard, outperforming MobileLLM-R1. Haidass-Translate-143M exceeds NLLB-200-distilled-600M on en→zh and approaches Qwen3-0.6B's bidirectional BLEU at roughly $1/4$ of its size.

2 HAIDASS: MODEL AND PRETRAINING

Haidass-143M is a decoder-only Chinese–English bilingual language model trained from scratch on 391B tokens. This section describes the architecture of the released model, the bilingual pretraining data, and the Natural-to-Synthetic Data Curriculum (i.e., the sequence of data mixtures).

2.1 MODEL ARCHITECTURE

We adopt a Qwen3-style Transformer (Yang et al., 2025) with 30 layers and a hidden size of 576. Each attention layer has 9 query heads and 3 key/value heads, with a head dimension of 64. The feed-forward network (FFN) uses SwiGLU with an intermediate size of 1,536 and Table 1 summarizes the configuration.

Table 1: Architecture of the released Haidass-143M.

Property	Value	Property	Value
Total parameters	143,071,296	Transformer layers	30
Hidden size	576	FFN intermediate size	1,536
Query / KV heads	9 / 3	Head dimension	64
Vocabulary size	64,000	Context length	4,096
RoPE base	100,000	RMSNorm ϵ	10^{-6}

Chinese and English share a custom 64,000-entry BPE vocabulary and the same model weights. The input embedding and output projection are tied, resulting in a single 36.86M-parameter embedding matrix and approximately 106.21M parameters elsewhere in the model. This architecture remains unchanged throughout training.

2.2 Bilingual Pretraining Data

The web-text component comes primarily from Ultra-FineWeb-en and Ultra-FineWeb-zh (Wang et al., 2025). We use Ultra-FineWeb-zh as the primary Chinese source. For English, we further filter the English pool with a fastText-based quality classifier, retaining scores of at least 0.9 to obtain English HQ. Ultra-FineWeb already performs upstream filtering; the additional English threshold is a further restriction within that corpus. DCLM supplies another English web-text source (Li et al., 2024b).

To extend the mixture beyond web text, we include Cosmopedia synthetic text and FineMath-4plus mathematical text. Later mixtures introduce the multi-style rewrites and question–answer data from Ultra-FineWeb-L3 (Lyu et al., 2026). These sources provide different forms of textual data within the same next-token prediction objective. The final mixture also includes Wikipedia-labelled text, PG-19 (Rae et al., 2019), WikiHow-derived data (Koupaee & Wang, 2018), and Chinese rewritten text. Appendix A lists the sources and configured sampling weights.

The data mixture varies across training phases rather than maintaining a fixed Chinese–English ratio throughout pretraining. The main Chinese web-text weight ranges from 18% to 35% in the first five phases. These weights describe each phase's sampling configuration; the independent experiments in Section 4 examine particular data choices under smaller budgets.

2.3 Training Recipe

We train Haidass-143M using MindSpeed-LLM on Ascend 910B, with a global batch size of 128 sequences and a sequence length of 4,096 tokens. Therefore, each optimizer update processes 524,288 tokens. The released checkpoint is taken at iteration 746,000, corresponding to 391B cumulative training tokens on successive resumed runs. Figure 2 summarizes the six training phases and the corresponding data mixtures.

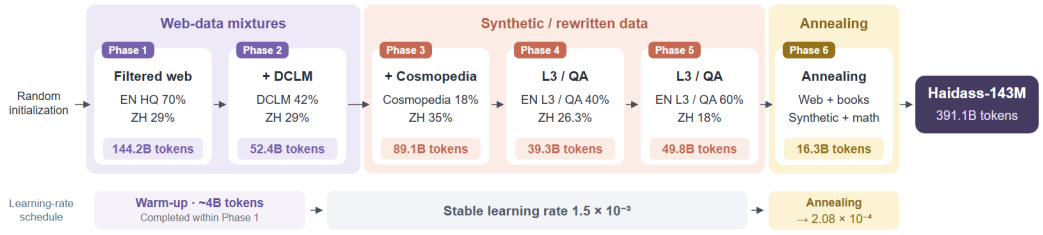

Figure 2: Pretraining Haidass-143M. The six boxes show successive training recipe. Complete weights, durations, and token counts are given in Appendix A.

The first two phases primarily use filtered web text. The first phase combines English HQ, Chinese web text, and FineMath-4plus; the second phase additionally incorporates DCLM. The third phase introduces synthetic text from Cosmopedia. In the fourth and fifth phases, the combined sampling

weight of English text and question–answer data from Ultra-FineWeb-L3 is set to 40% and 60%, respectively, while Chinese web text and other data sources remain in the mixture. The final phase comprises approximately 16.3B tokens and uses the broader mixture described in the preceding subsection, together with learning-rate annealing.

The learning rate warms up over the first 7,629 updates, approximately 4B tokens, and remains at 1.5×10^{-3} until final annealing. Then it decreases to 2.08×10^{-4} at the released checkpoint. The data-mixture schedule is decoupled from the learning rate schedule: warm-up is completed within the first data phase, phases two through five are entered at a constant learning rate, and annealing begins with the sixth phase. The annealing mixture incorporates two additional sources: `benchmark_train` (0.77%), consisting of public benchmark training splits with overlap cleaning, and `hs_syn` (1.85%). More details are given in Appendix A.

3 Model Capabilities and Task Adaptation

We first evaluate Haidass-143M on English and Chinese benchmarks, then assess its effectiveness as an initialization for supervised bidirectional translation.

3.1 Evaluation Setup

For English-language evaluation, we use the Open SLM Leaderboard snapshot dated 22 September 2026. The leaderboard reports scores on HellaSwag, ARC-Easy, ARC-Challenge, PIQA, and ArithMark-3, with the last evaluating arithmetic through English multiple-choice questions. We use the published task scores and the leaderboard's aggregate Intelligence Index. The snapshot contains 185 scored entries, including models with different architectures and training configurations; its index definition and complete comparison details are given in Appendix B.1.

We evaluate Chinese reading comprehension using C3 (Sun et al., 2020), Chinese knowledge using CMMLU (Li et al., 2024a), and reading comprehension in both languages using the Chinese and English subsets of Belebele (Bandarkar et al., 2024). All seven released bases are evaluated on these benchmarks, together with SocialIQA (1,954 examples) and CEval-valid (1,346). ARC-Easy, ARC-Challenge, PIQA, and OpenBookQA are retained as supplementary results in Appendix B.2.1. All evaluations use zero-shot candidate-answer likelihoods without chat templates. The aggregate results are taken from a fixed evaluation archive; protocol details and length-normalized results are given in Appendix B.2.

3.2 Base-Model Capabilities

English-language performance. Table 2 reports English-language performance on the Open SLM Leaderboard. Haidass-143M achieves an index of 25.07, ranking 5th; the 4th- and 6th-ranked entries score 25.17 and 24.96, respectively. Compared with SmolLM2-135M, Haidass-143M scores lower on all four non-arithmetic benchmarks and 0.40 percentage points higher on ArithMark-3.

Table 2: Top ten entries on the Open SLM Leaderboard. H/AE/AC/P/A3 denote HellaSwag, ARC-Easy, ARC-Challenge, PIQA, and ArithMark-3.

#	Model	Params	Index	H	AE	AC	P	A3
1	SmolLM2-135M	135.00M	27.13	43.22	58.63	29.69	68.44	39.20
2	cagliostro-v3	146.35M	26.55	42.51	54.88	28.75	67.46	43.70
3	SmolLM-135M	135.00M	25.74	42.70	56.31	29.01	68.28	36.80
4	GPT-X2.5-135M	135.03M	25.17	40.57	51.81	29.18	69.42	38.40
5	**Haidass-143M**	**143.07M**	**25.07**	**40.58**	**56.36**	**27.39**	**67.79**	**39.60**
6	BananaMind-2-Pro	138.97M	24.96	42.78	53.58	27.82	67.52	38.20
7	MobileLLM-R1-140M-base	140.00M	24.64	33.84	49.92	24.74	63.22	65.70
8	GPT-X2-125M	125.00M	23.36	40.41	51.47	27.82	67.30	37.20
9	Flare	130.39M	22.49	39.43	51.14	26.88	65.94	39.20
10	Speck2-140M-Instruct	140.65M	21.33	38.94	47.73	25.34	65.83	39.10

Bilingual capability evaluation. Table 3 complements the English leaderboard with Chinese and bilingual evaluation. Among the five models below 200M parameters, Haidass-143M achieves the highest C3 accuracy at 41.96%, followed by GPT-X2.5-135M (38.10%), SmolLM2-135M (37.82%), cagliostro-v3 (37.67%), and MobileLLM-R1-140M-base (33.89%). Against SmolLM2, the 4.14-point difference has a paired passage-bootstrap 95% interval of [2.51, 5.77] points. Length-normalized scoring preserves Haidass's lead at 44.42%. The five compact models remain near the four-choice chance level on CMMLU, C-Eval, and both Belebele subsets, and occupy a narrow 38.18–40.79% range on SocialIQA. The larger Qwen bases lead the Chinese and bilingual benchmarks.

Table 3: Accuracy and text likelihood of Haidass-143M and other SLMs. B-ZH/B-EN denote the Chinese/English Belebele subsets; F/W denote FLORES-200 dev/Wikipedia.

		Accuracy (%, ↑)						BPB (↓)			
Model	Params	C3	CMMLU	B-ZH	B-EN	SIQA	C-Eval	F-ZH	F-EN	W-ZH	W-EN
Haidass-143M	**143M**	**41.96**	**25.44**	**22.89**	**24.44**	**38.43**	**25.71**	**1.48**	**1.21**	**1.50**	**0.97**
SmolLM2-135M	135M	37.82	24.89	22.67	24.22	39.46	25.41	2.04	1.08	1.91	0.91
GPT-X2.5-135M	135M	38.10	25.27	22.89	26.44	38.18	22.88	2.78	1.10	2.50	0.94
cagliostro-v3	146M	37.67	25.26	26.00	27.11	40.79	22.88	2.85	1.11	2.52	0.96
MobileLLM-R1-140M-base	140M	33.89	25.28	24.33	25.22	39.87	24.37	2.34	1.25	2.15	1.04
Qwen2.5-0.5B	494M	48.97	48.80	44.44	45.78	43.86	52.15	1.35	1.07	1.21	0.87
Qwen3-0.6B-Base	596M	52.42	51.18	60.67	56.56	43.30	50.15	1.36	1.10	1.12	0.86

Table 3 also reports text likelihood. On Chinese text from FLORES-200 dev and Wikipedia, Haidass-143M achieves lower BPB than all other models below 200M parameters. SmolLM2-135M, GPT-X2.5-135M, and cagliostro-v3 obtain lower English BPB than Haidass on both corpora. Appendix B.2.1 gives the evaluation protocol and supplementary task results.

3.3 Adaptation to Bidirectional Translation

We use translation to assess the value of the base as a task-specific initialization. The current Haidass-Translate-143M release starts from Haidass1.5-143M and is trained in three stages. Stage 1 performs translation-only supervised fine-tuning on 7.84M cleaned Chinese–English sentence pairs, expanded to 15.67M bidirectional samples; its packed sequences isolate inter-document attention, and the checkpoint is selected at 4.50 epochs on FLORES-200 dev. Stage 2 anneals on 715,658 curated pairs after removing noisy or contaminated sources and adding selected corpora. Stage 3 applies sequence-level distillation from Qwen3.8-27B using 1,579,486 samples, with 70% teacher-generated targets and 30% original corpus data. Appendix F gives the full recipe.

We report corpus BLEU and chrF++ on FLORES-200 dev (997 sentences per direction) and FLORES+ devtest (1,012 sentences per direction). Model selection and direction-specific beam parameters are frozen on dev before a single evaluation of each decoding configuration on devtest. The released generation configuration is greedy decoding. The beam configuration uses three beams with length penalty 1.25 for en→zh and four beams with length penalty 0.90 for zh→en; both are deterministic. All other models in Table 4 use greedy decoding.

Relative to Drafter-143M, greedy Haidass initialization raises en→zh BLEU from 10.93 to 29.23 and zh→en BLEU from 5.82 to 17.50 (Table 4); the corresponding chrF++ gains are 10.90 and 17.18 points. With architecture, data, and training recipe matched, this result shows that Haidass pretraining provides a strong initialization for bidirectional translation.

Beam search improves the released model to 29.81/21.02 BLEU/chrF++ for en→zh and 18.69/45.80 for zh→en on devtest. It trails Qwen3-0.6B by 1.95 and 0.97 BLEU in the two directions while using approximately 24% as many parameters, and exceeds Qwen2.5-0.5B-Instruct by 0.49 and 0.65 BLEU. The NLLB reference remains stronger for zh→en. On FLORES-200 dev, the same beam configuration reaches 30.36/21.40 and 20.62/46.51, respectively.

We audit all 15.83M training samples for overlap with both test sets using consecutive eight-word English or ten-character Chinese fragments. The audit finds 1,147 hits (0.0072%) against FLORES-200 dev and 1,788 (0.0113%) against FLORES+ devtest. Manual inspection identifies common-

Table 4: Bidirectional translation results on FLORES-200 dev and FLORES+ devtest.

Model	Params	FLORES-200 dev				FLORES+ devtest			
		EN–ZH		ZH–EN		EN–ZH		ZH–EN	
		BLEU	chrF++	BLEU	chrF++	BLEU	chrF++	BLEU	chrF++
HY-MT1.5-1.8B	1800M	44.65	30.98	27.68	57.96	37.36	26.08	20.33	51.48
Qwen3-0.6B	600M	30.94	21.10	20.21	48.62	31.76	21.48	19.66	48.14
Haidass-Translate-143M-beam	**143M**	**30.36**	**21.40**	**20.62**	**46.51**	**29.81**	**21.02**	**18.69**	**45.80**
Qwen2.5-0.5B-Instruct	500M	28.96	19.65	18.09	45.85	29.32	19.95	18.04	46.00
Haidass-Translate-143M	**143M**	**28.32**	**19.49**	**17.72**	**44.32**	**29.23**	**19.90**	**17.50**	**44.01**
M2M-100-418M	418M	28.04	20.53	20.58	48.79	28.29	20.60	19.52	47.87
NLLB-200-distilled-600M	600M	22.44	16.74	25.71	52.28	23.07	16.94	24.30	51.48
Drafter-143M (random init.)	143M	12.04	9.43	5.47	27.31	10.93	9.00	5.82	26.83

phrase fragments rather than full-sentence leakage. Stage 2 and Stage 3 use the same audited sources and filtering rule. These results establish the current release-level translation scores; they replace the earlier two-epoch development values.

4 Data Curation and Language Scheduling

We study how training-data selection and language mixing affect Haidass-143M training. We first assess quality filtering across model scales, then examine language ordering at matched per-language token budgets and language allocation at a fixed total token budget. To isolate the effects of individual data and training choices, all models in this section are trained from random initialization rather than initialized from the released Haidass-143M checkpoint, and follow individual training-data choices rather than the full six-phase training recipe described in Section 2. The 143M models share the architecture and tokenizer of the released model, while other scales use scaled configurations. All models adopt the architecture and tokenizer of the released model, scaled to different parameter sizes.

4.1 Quality Filtering Across Model Scales

To assess the effect of additional quality filtering on English pretraining, we compare two data conditions at five model sizes: 56M, 143M, 263M, 469M, and 803M. Full uses the available Ultra-FineWeb-en corpus, whereas HQ retains only documents with fastText-based quality scores of at least 0.9. Each model trains for approximately 20B . We evaluate the final checkpoints on ten zero-shot tasks using Lighteval and compute the macro-average accuracy under both standard candidate scoring (`acc`) and length-normalized candidate scoring (`acc_norm`). Appendix C provides the training configurations, evaluation tasks, and metric definitions.

Table 5: Final ten-task macro scores at 20B tokens. Δ is HQ minus Full in pp.

	`acc`			`acc_norm`		
Size	Full	HQ	Δ	Full	HQ	Δ
56M	39.00	40.05	+1.06	40.44	41.21	+0.77
143M	42.34	42.74	+0.39	43.28	44.03	+0.75
263M	43.96	44.81	+0.85	45.07	45.97	+0.90
469M	45.51	46.54	+1.03	46.60	47.83	+1.23
803M	46.60	47.44	+0.84	47.91	48.92	+1.01

The HQ condition yields higher final macro-average accuracy than Full at all five model sizes under both scoring methods (Table 5). The gains range from 0.39 to 1.06 percentage points for `acc` and from 0.75 to 1.23 percentage points for `acc_norm`.The gain is non-monotonic in model size: 143M has the smallest raw-accuracy gain, while 469M has the largest normalized-accuracy gain. At 143M, an intermediate threshold of 0.7 yields lower `acc` but higher `acc_norm` than Full, showing that a higher filtering threshold does not necessarily improve all aggregate metrics (Appendix C.3).

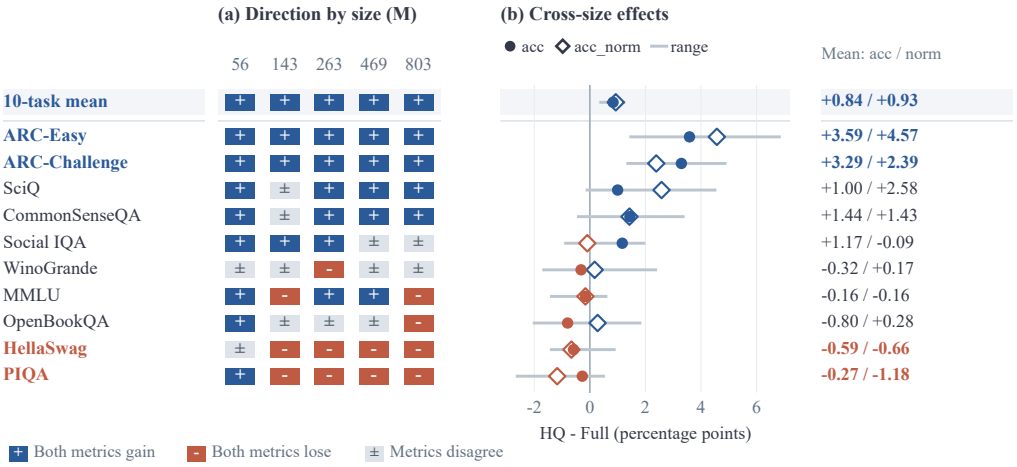

Figure 3: **Aggregate gains coexist with task-level losses.** (a) Signs summarize agreement between `acc` and `acc_norm` at each model size. (b) Markers and paired values show the mean HQ–FULL differences across the five model sizes for `acc` and `acc_norm`, respectively. Gray segments show the range across the ten size–metric combinations.

Figure 3 shows that higher average accuracy does not imply improvements on every task. HQ improves accuracy on ARC-Easy and ARC-Challenge at every model size under both scoring methods. In contrast, HQ reduces accuracy on HellaSwag and PIQA across all evaluated scales. At 469M, for example, the `acc` gains are 6.44 and 3.67 percentage points on ARC-Easy and ARC-Challenge, respectively, while the corresponding losses are 1.11 points on HellaSwag and 0.54 points on PIQA. These results reveal a trade-off: the aggregate benefit of this filtering policy comes with task-level performance variations.

4.2 Language Ordering and Allocation

Language ordering. To isolate the effect of language ordering, we vary the sequence of Chinese and English exposure while holding the total number of tokens in each language fixed within each model size. We compare sequence-level independent and identically distributed (IID) mixing at a 50:50 language ratio, eight alternating monolingual training phases, and a single switch from English to Chinese or from Chinese to English. The 43M- and 143M-parameter models below are trained from random initialization on 100M and 331M tokens. For each language, we compute the relative change in held-out packed-block loss from the IID baseline and average the changes across the two languages. Appendix D defines the loss metric and provides further analysis of held-out loss over the course of training.

Random mixing gives the lowest balanced loss at both scales (Figure 4b). Eight-block alternation is closer to IID than a single switch, but still incurs penalties of 4.57% and 4.62%. The one-switch schedules strongly favor the language seen last. At 143M, training on English and then Chinese raises English loss by 38.22% relative to IID, whereas Chinese loss rises by only 1.25%. Reversing the order raises Chinese loss by 74.02%, while English loss is 1.21% below IID.

A separate 143M study varies the language ratio at a fixed 1B-token budget (Figure 4a). All three bilingual mixtures have higher PPL than each language's own monolingual reference. At 50:50, the English and Chinese increases are 17.32% and 19.75%; shifting the ratio favors the more frequent language. The full five-ratio results and protocol are in Appendix E.

Together, these comparisons show that both language allocation and input order affect bilingual retention. The ratio study changes each language's training exposure, whereas the order study holds language totals fixed within each scale. The short-budget results favor continuous language interleaving and motivate tuning the allocation with longer runs and a finer block-length sweep.

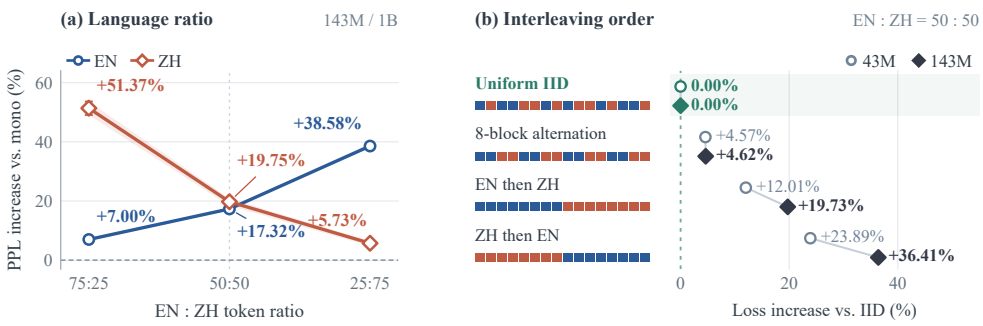

Figure 4: Effects of language allocation and ordering. (a) Relative PPL increase over monolingual baselines for 143M models trained on 1B tokens across English–Chinese ratios.(b) Mean relative increase in held-out packed-block loss over IID mixing for different language-order schedules at a fixed 50:50 token ratio.

5 Related Work

The development of small language models (SLMs) has motivated research on both architectural efficiency and training-data design. MobileLLM explores architectures tailored to the sub-billion-parameter regime, highlighting the benefits of deep and thin networks, embedding sharing, and grouped-query attention (Liu et al., 2024). Alongside architectural advances, data-centric approaches have examined how corpus quality and composition affect model capabilities. DataComp-LM and FineWeb investigate filtering and deduplication for pretraining data, while Ultra-FineWeb extends efficient quality filtering to English and Chinese web corpora (Li et al., 2024b; Penedo et al., 2024; Wang et al., 2025). Beyond selecting individual documents, DoReMi and RegMix optimize the proportions of different data sources using proxy-model training and performance prediction, respectively (Xie et al., 2023; Liu et al., 2025). More recently, MobileLLM-R1 studies capability-aware data curation and reweighting to develop reasoning capabilities in sub-billion-parameter models (Zhao et al., 2026). Together, these studies establish data selection and allocation as important dimensions of small-model development.

Synthetic data provides another means of shaping the pretraining distribution. WRAP combines natural web text with synthetic rephrasings in different styles, including question–answer formats, to improve pretraining efficiency (Maini et al., 2024). Ultra-FineWeb-L3 provides Chinese and English training data through multi-style rewriting and question–answer generation (Lyu et al., 2026). Beyond constructing these data sources, their use also involves deciding when and how much to sample during training. SmolLM2-1.7B adopts a multi-stage mixture of web text and specialized data, whereas its 135M and 360M variants use a single-stage data recipe (Ben Allal et al., 2025). These studies motivate examining both the composition of pretraining data and its allocation across training stages.

Bilingual pretraining introduces the additional challenge of supporting two languages within a shared parameter budget. Research on multilingual models has examined the trade-off between positive cross-lingual transfer and capacity dilution (Conneau et al., 2020). Controlled language-modeling experiments further show that the effects of multilingual training depend on data availability, linguistic similarity, and model size (Chang et al., 2024). Cross-lingual continual-learning studies also examine how sequential exposure to languages affects knowledge retention (M'hamdi et al., 2023; Winata et al., 2023). Our work investigates data curation and training schedules for Chinese–English bilingual SLMs through controlled comparisons of quality filtering, language allocation, and language ordering. We further develop a Natural-to-Synthetic Data Curriculum for Haidass-143M, beginning predominantly with filtered natural text, subsequently introducing synthetic text and increasing the emphasis on rewritten and question–answer data, followed by a final annealing stage.

6 CONCLUSION

We present Haidass, a Chinese-English bilingual small language model family with Haidass-143M ranking 5th on the Open SLM Leaderboard. We explore data curation and training schedules under a limited parameter budget. This curriculum begins predominantly with filtered natural text, then introduces synthetic text and places increasing emphasis on rewritten and question–answer data, and concludes with a final annealing stage. Trained from scratch on 391.1B tokens using Ascend 910B, Haidass-143M achieves competitive English benchmark performance and improves Chinese reading comprehension on C3 over the similarly sized SmolLM2-135M. It also provides an effective initialization for supervised bidirectional translation, enabling Haidass-Translate-143M to achieve competitive translation quality relative to larger reference models. Experiments further show that quality filtering improves average benchmark scores while introducing task-level trade-offs, and that random language interleaving better preserves bilingual performance than the tested block schedules at matched per-language token budgets. Language allocation separately governs the trade-off between English and Chinese perplexity under a fixed total token budget. Together, these results provide an effective for data selection and language scheduling in bilingual SLMs, alongside a concrete recipe for integrating natural and synthetic pretraining data.

AI USE STATEMENT

Generative AI tools, including ChatGPT , were used to assist with language polishing and LaTeX formatting. The authors take full responsibility for reviewing and verifying all scientific content, mathematical formulations, experimental configurations, numerical results, and conclusions, and for the final manuscript and all reported results.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. URL https://arxiv.org/abs/2303.08774.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. The Belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 749–775, 2024. doi: 10.18653/v1/2024.acl-long.44. URL https://aclanthology.org/2024.acl-long.44/.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, et al. SmolLM2: When smol goes big – data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*, 2025. URL https://arxiv.org/abs/2502.02737.

Tyler A. Chang, Catherine Arnett, Zhuowen Tu, and Benjamin K. Bergen. When is multilinguality a curse? language modeling for 250 high- and low-resource languages. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4074–4096, 2024. URL https://aclanthology.org/2024.emnlp-main.236/.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8440–8451. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.747. URL https://aclanthology.org/2020.acl-main.747/.

Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024. URL https://arxiv.org/abs/2404.06395.

Aishwarya Kamath, Johan Ferret, Shreya Pathak, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. URL https://arxiv.org/abs/2503.19786.

Mahnaz Koupaee and William Yang Wang. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*, 2018. URL https://arxiv.org/abs/1810.09305.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. CMMLU: Measuring massive multitask language understanding in Chinese. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11260–11285, 2024a. doi: 10.18653/v1/2024.findings-acl.671. URL https://aclanthology.org/2024.findings-acl.671/.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, et al. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282, 2024b.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. RegMix: Data mixture as regression for language model pre-training. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=5BjQOUXq7i.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020.

Zechun Liu, Changsheng Zhao, Forrest Iandola, et al. MobileLLM: Optimizing sub-billion parameter language models for on-device use cases. In *International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2402.14905.

Hongya Lyu, Chuyue Zhou, Shixin Ren, et al. Ultra-FineWeb-L3, 2026. URL https://huggingface.co/datasets/openbmb/Ultra-FineWeb-L3.

Pratyush Maini, Skyler Seto, Richard Bai, David Grangier, Yizhe Zhang, and Navdeep Jaitly. Rephrasing the web: A recipe for compute and data-efficient language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14044–14072. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.757. URL https://aclanthology.org/2024.acl-long.757/.

Meryem M'hamdi, Xiang Ren, and Jonathan May. Cross-lingual continual learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3908–3943, 2023. URL https://aclanthology.org/2023.acl-long.217/.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, et al. The FineWeb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024. URL https://arxiv.org/abs/2406.17557.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019. URL https://arxiv.org/abs/1911.05507.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. Investigating prior knowledge for challenging Chinese machine reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:141–155, 2020. doi: 10.1162/tacl_a_00305. URL https://aclanthology.org/2020.tacl-1.10/.

Yudong Wang, Zixuan Fu, Jie Cai, et al. Ultra-FineWeb: Efficient data filtering and verification for high-quality llm training data, 2025. URL https://arxiv.org/abs/2505.05427.

Genta Winata, Lingjue Xie, Karthik Radhakrishnan, Shijie Wu, Xisen Jin, Pengxiang Cheng, Mayank Kulkarni, and Daniel Preotiuc-Pietro. Overcoming catastrophic forgetting in massively multilingual continual learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 768–777, 2023. URL https://aclanthology.org/2023.findings-acl.48/.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. DoReMi: Optimizing data mixtures speeds up language model pretraining. *arXiv preprint arXiv:2305.10429*, 2023. URL https://arxiv.org/abs/2305.10429.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Changsheng Zhao, Ernie Chang, Zechun Liu, Chia-Jung Chang, Wei Wen, Chen Lai, Rick Cao, Yuandong Tian, Raghuraman Krishnamoorthi, Yangyang Shi, and Vikas Chandra. MobileLLM-R1: Exploring the limits of sub-billion language model reasoners with open training recipes. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.24945.

A Pretraining Data Mixtures and Schedule

This appendix supplements the pretraining curriculum described in the main text. Table 6 details the six data-mixture phases shown in Figure 2, including their update ranges, token budgets, and source compositions. The data-mixture phases are defined separately from the learning-rate schedule: warm-up occurs within Phase 1, Phases 2–5 use aconstant learning rate, and annealing begins in Phase 6.

Table 6: Data mixtures and token budgets for the six pretraining phases.

Phase	Updates (k)	Tokens (B)	Data mixture
1	0–275	144.18	English HQ 70%, Chinese web text 29%, FineMath-4plus 1%
2	275–375	52.43	DCLM 42%, Chinese web text 29%, English HQ 28%, FineMath-4plus 1%
3	375–545	89.13	Chinese web text 35%, English HQ 25.2%, Cosmopedia 18%, DCLM 16.8%, FineMath-4plus 5%
4	545–620	39.32	English L3/QA 40%, Chinese web text 26.3%, English HQ 18.4%, DCLM 12.3%, FineMath-4plus 3%
5	620–715	49.81	English L3/QA 60%, Chinese web text 18%, DCLM 12%, English HQ 8%, FineMath-4plus 2%
6	715–746	16.25	English HQ, DCLM, reference and book text, synthetic and rewritten text, and mathematical text

English HQ denotes Ultra-FineWeb-en documents with fastText quality scores of at least 0.9, and Chinese web text denotes Ultra-FineWeb-zh. English L3/QA comprises English rewritten text and question–answer data from Ultra-FineWeb-L3.

Training-token budget. Each optimizer update processes 128 sequences of 4,096 tokens, corresponding to 524,288 tokens per update. The six phases span 746,000 updates in total, yielding approximately 391.1B training tokens.

Annealing mixture. Phase 6 combines English HQ (25%), DCLM (22.8%), a Wikipedia-labelled corpus (12%), PG-19 (8.4%), Cosmopedia-WikiHow (6.76%), WikiHow-QA (6.18%), FineMath-4plus (5%), MiniMind text-to-text data (4%), and Chinese L3 (4%), together with smaller rewritten-text, story, and benchmark-related components. The components identified as `benchmark_train` and `hs_syn` have sampling weights of 0.77% and 1.85%, respectively.

Learning-rate annealing. During Phase 6, the learning rate decreases from approximately 1.499×10^{-3} to 2.080×10^{-4} at the final checkpoint.

B Evaluation Protocols and Supplementary Results

This appendix provides the evaluation protocols and supplementary results for the SLMs comparisons in the main text. We first describe the Open SLM Leaderboard and its ranking metric, then detail our evaluation of Chinese and bilingual comprehension, additional benchmark tasks, and text likelihood.

B.1 Leaderboard Protocol and Task-Level Comparisons

Evaluated model and comparison set. We use the Open SLM Leaderboard snapshot dated 22 September 2026. Table 2 presents its ten highest-ranked entries without additional filtering. The snapshot contains 185 scored entries: 176 with scores on all five tasks and nine evaluated under the leaderboard's rules for missing components. The Haidass-143M entry corresponds to the model trained on 391.1B tokens described in Section 2. All leaderboard comparisons use the published scores.

Evaluation protocol and ranking metric. The leaderboard focuses on models with approximately 150M parameters or fewer. It reports zero-shot results on HellaSwag, ARC-Easy, ARC-Challenge, and PIQA using lm-evaluation-harness, together with results on ArithMark-3. ArithMark-3 comprises 1,000 four-choice English arithmetic continuation questions scored by answer likelihood. We retain the task-scoring conventions used by the leaderboard.

For a task score s expressed as a percentage and a chance-level accuracy c, the chance-adjusted score is

$$g(s, c) = 100\frac{s - c}{100 - c}.$$

Let s_H, s_E, s_C, s_P, and s_A denote the scores on HellaSwag, ARC-Easy, ARC-Challenge, PIQA, and ArithMark-3, respectively. For entries with scores on all five tasks, the Intelligence Index is

$$I = \frac{g(s_H, 25) + g((s_E + s_C)/2, 25) + g(s_P, 50) + 0.65\, g(s_A, 25)}{3.65}. \tag{1}$$

ARC-Easy and ARC-Challenge contribute through their mean, while ArithMark-3 has a relative weight of 0.65. ArithMark-2 is not included. All ten entries in Table 2 have complete five-task scores, so Equation 1 applies to each of them. The length normalization of candidate-answer scores is denoted by `acc_norm` in the quality-filtering experiments.

Ranking and task-level differences. Haidass-143M ranks fifth with an Intelligence Index of 25.07, 0.10 points below the fourth-ranked entry. Figure 1 places these results in the context of the full leaderboard.

The small differences in aggregate scores conceal larger differences on individual tasks. Compared with the adjacent GPT-X2.5-135M entry, Haidass-143M scores 4.55 percentage points higher on ARC-Easy and 1.20 points higher on ArithMark-3, but 1.79 points lower on ARC-Challenge and 1.63 points lower on PIQA. Their HellaSwag scores differ by only 0.01 points. Thus, a similar Intelligence Index does not imply similar performance across tasks.

The comparison with MobileLLM-R1-140M-base illustrates a different trade-off. Haidass-143M scores higher on HellaSwag, ARC-Easy, ARC-Challenge, and PIQA by 6.74, 6.44, 2.65, and 4.57 percentage points, respectively, but scores 26.10 points lower on ArithMark-3. Under the leaderboard's weighting scheme, these differences yield an overall index advantage of approximately 0.43 points.

Scope of the comparison. The leaderboard measures performance on English-language benchmarks, including arithmetic; Chinese and bilingual comprehension are evaluated separately in Section 3.2. The compared models differ in training-token budgets, tokenizers, architectures, and post-training procedures. For example, Speck2-140M-Instruct is instruction-tuned, whereas Flare includes recurrent layers. The leaderboard therefore provides a comparison of model performance rather than a controlled estimate of the effect of any individual training choice.

B.2 Released-Base Evaluation Protocol and Supplementary Results

Evaluation setup. Released bases are evaluated with lm-evaluation-harness 0.4.14.dev0 (a source snapshot without Git metadata), Transformers 4.57.1, Tokenizers 0.22.2, PyTorch/torch_npu 2.7.1, and BF16 on Ascend 910C. Evaluation is zero-shot and uses candidate-answer likelihoods without chat templates or chain-of-thought prompting. Belebele uses English instructions for both languages. The fixed archive contains C3, CMMLU, and Belebele records for all seven released bases; their core scores, aggregate SocialIQA and C-Eval results, and BPB results appear in Table 3.

C3 comprises 1,890 dialogue questions and 2,002 mixed-text questions, combined by question count. CMMLU contains 11,582 questions, and each Belebele language subset contains 900. The archive contains 120,918 records. Of the 3,892 cagliostro-v3 C3 examples, 84 (2.2%) reach its 2,048-token context limit; all other model–task combinations report no truncation. We verified that raw predictions equal the argmax of the archived candidate log likelihoods, correctness matches the predicted and gold options, sample identities and gold answers align across models, and recomputed raw and normalized aggregates match the published summaries to their reported precision.

Aggregation and uncertainty estimation. Accuracy is computed from per-example predictions. For the C3 accuracy difference between Haidass-143M and SmolLM2-135M, we use a paired bootstrap over passages to account for questions that share a passage. Each bootstrap replicate resamples passage groups with replacement separately within the dialogue and mixed-text subsets, which contain 1,627 and 1,045 passage groups, respectively. The same sampled groups are used for both models, and accuracy is recomputed with equal weight per question. The resulting interval quantifies benchmark-sampling uncertainty under paired passage resampling.

Reproduction artifacts. The frozen archive supports score recomputation from the exported records. Inference-level reproduction additionally uses the task YAML files, inference scripts, exact templates, candidate normalization, and model-weight and dataset checksums referenced by the report. The report describes letter-only candidates for all tasks, while C3 raw and length-normalized predictions differ and the exports omit candidate strings and lengths; we therefore report both C3 metrics.

B.2.1 Supplementary Tasks and Text Likelihood

Additional benchmark tasks. Table 7 reports the remaining zero-shot results on ARC-Easy, ARC-Challenge, PIQA, and OpenBookQA. These results come from the base-model evaluation described above, separately from the published leaderboard scores in Table 2. Only aggregate scores are available for these supplementary tasks, so the per-example consistency checks for C3, CMMLU, and Belebele do not extend to this table.

Table 7: Supplementary zero-shot accuracy (%). Paired entries report accuracy under raw and length-normalized candidate scoring.

Model	ARC-E	ARC-C	PIQA	OBQA
Haidass-143M	**59.22/54.71**	**25.85/28.07**	**68.34/68.12**	**18.00/32.00**
SmolLM2-135M	64.44/58.80	28.07/29.86	68.28/68.50	21.60/33.20
GPT-X2.5-135M	58.16/52.48	25.94/28.75	67.46/69.21	22.60/31.20
cagliostro-v3	60.02/54.92	26.19/28.75	67.57/67.68	22.20/31.60
MobileLLM-R1-140M-base	53.28/49.83	21.25/24.15	64.25/63.22	15.40/29.20
Qwen2.5-0.5B	64.56/58.88	28.92/32.34	70.62/70.13	25.00/34.80
Qwen3-0.6B-Base	65.49/57.62	33.79/38.31	69.59/69.80	25.00/35.40

Text-likelihood evaluation. To compare text likelihood across models with different tokenizers, we report bits per UTF-8 byte (BPB):

$$\mathrm{BPB} = \frac{\mathrm{NLL}}{B \ln 2},$$

where NLL is the total negative log-likelihood in natural-log units and B is the UTF-8 byte count used for normalization.

We evaluate the Chinese and English subsets of FLORES-200 dev, with 997 sentences per language, and samples from Wikipedia, with 2,000 documents per language. The evaluation uses nonoverlapping chunks of 1,024 tokens and adds a beginning-of-sequence token when required by the evaluation implementation. The byte totals used for normalization are identical across models within each corpus.

Byte normalization provides a common denominator across tokenizers. However, a fixed context length in tokens can correspond to different amounts of text, and chunk boundaries can differ across tokenizers. The BPB comparison is therefore conditional on this evaluation protocol.

As shown in Table 3, Haidass-143M achieves the lowest Chinese BPB on both corpora among the five similarly sized models in the upper group. SmolLM2-135M, GPT-X2.5-135M, and cagliostro-v3 achieve lower English BPB than Haidass-143M on both corpora. Both larger Qwen models achieve lower BPB than Haidass-143M in both languages, although neither has the lowest BPB across all four evaluations.

C Quality-Filtering Settings and Additional Analysis

This appendix supplements the quality-filtering analysis in the main text with training configurations, a comparison of filtering thresholds, task-level results, and performance trajectories. The experiments use English web text and approximately 20B training tokens per run.

C.1 Experimental Setup

Model configurations and training. We evaluate five Qwen3-style configurations with approximately 56M, 143M, 263M, 469M, and 803M parameters (Table 8). All configurations use RMSNorm, SwiGLU, RoPE, grouped-query attention, a shared 64,000-entry vocabulary, and tied input and output embeddings. At each model size, the FULL and HQ conditions use the same architecture and training hyperparameters. An additional filtering threshold is evaluated at 143M, giving eleven training conditions in total.

Table 8: Model configurations for the quality-filtering experiments.

Total	Non-embedding	Hidden size	Layers	Heads	FFN size
56.0	31.5	384	20	6/2	1024
143.1	106.2	576	30	9/3	1536
263.1	214.0	768	34	12/4	2048
469.0	403.5	1152	28	18/6	3072
803.0	704.7	1536	28	24/8	4096

All runs use a sequence length of 4,096 and a batch size of 128 sequences. The optimizer is AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$, weight decay 10^{-5}, and gradient clipping at 1.0. The warm-up–stable–decay schedule uses a peak learning rate of 1.5×10^{-3}, approximately 2B tokens for warm-up, and approximately 4B tokens for decay to zero. Final performance is evaluated after 38,146 optimizer updates, corresponding 20B training tokens. Training-token budgets are matched across model sizes, whereas computational costs increase with model size.

Data conditions. The FULL condition uses the available Ultra-FineWeb-en corpus without additional quality filtering. It retains the filtering already applied during construction of Ultra-FineWeb-en and therefore does not represent unfiltered web text. The HQ condition retains documents with fastText quality scores of at least 0.9.

Evaluation metrics and aggregation. We use Lighteval to evaluate ten tasks in a zero-shot setting: MMLU, ARC-Easy, ARC-Challenge, HellaSwag, PIQA, Social IQA, WinoGrande, OpenBookQA, CommonSenseQA, and SciQ. We report accuracy under raw candidate scoring (`acc`) as the primary metric and accuracy under length-normalized candidate scoring (`acc_norm`) to assess sensitivity to the scoring procedure.

For scoring metric m and task set $\mathcal{T}$, we define

$$\begin{aligned} S^m_{\mathcal{T}}(N, D, Q) &= \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} s^m_t(N, D, Q), \\ \Delta^m_{\mathcal{T}}(N, D) &= 100 \left[S^m_{\mathcal{T}}(N, D, \mathrm{HQ}) - S^m_{\mathcal{T}}(N, D, \mathrm{Full}) \right], \end{aligned} \tag{2}$$

where N is the number of model parameters, D is the cumulative number of training tokens, Q denotes the data condition, and s^m_t is task accuracy expressed as a fraction. Thus, $\Delta^m_{\mathcal{T}}$ is the HQ minus FULL difference in percentage points (pp).

The primary aggregate, S^m_{10}, assigns equal weight to all ten tasks. We additionally report S^m_4, the equal-weight mean over ARC-Easy, ARC-Challenge, HellaSwag, and PIQA, to examine sensitivity to task selection. Neither aggregate is weighted by evaluation-set size.

C.2 Final Aggregate Performance

As reported in Table 5, HQ achieves higher final ten-task mean accuracy than FULL at all five model sizes under both scoring procedures. The gains range from 0.39 to 1.06 pp for `acc` and from 0.75 to 1.23 pp for `acc_norm`.

The magnitude of the gain does not vary monotonically with model size. The smallest `acc` gain occurs at 143M, whereas the largest `acc_norm` gain occurs at 469M. These results establish a consistent direction of the final aggregate difference across the evaluated configurations, but do not establish a monotonic relationship between model size and filtering benefit.

C.3 Effect of the Filtering Threshold

At 143M, we additionally evaluate a fastText score threshold of 0.7, keeping the other training settings unchanged. Table 9 compares this condition with FULL and HQ after approximately 20B training tokens.

Relative to FULL, the 0.7 threshold improves the four-task mean under both scoring procedures. Its effect on the ten-task mean depends on scoring: accuracy under raw scoring decreases from 42.34% to 42.21%, whereas accuracy under length-normalized scoring increases from 43.28% to 43.52%. The 0.9 threshold achieves the highest mean in all four task-set–scoring combinations among the tested conditions. Thus, the measured effect of filtering depends on both the threshold and the evaluation criterion.

Table 9: Filtering-threshold comparison for the 143M model .

Condition	Ten-task acc	Ten-task acc_norm	Four-task acc	Four-task acc_norm
FULL	42.34	43.28	44.90	45.03
Score ≥ 0.7	42.21	43.52	45.33	45.94
Score ≥ 0.9	42.74	44.03	46.48	46.71

C.4 Task-Dependent Benefits and Trade-offs

Figure 3 shows that the aggregate gains are accompanied by substantial task-level differences. HQ improves ARC-Easy and ARC-Challenge at all five model sizes under both scoring procedures. At 469M, the gains are 6.44 and 3.67 pp under `acc`, and 6.82 and 3.16 pp under `acc_norm`, respectively.

For HellaSwag and PIQA, HQ yields lower final accuracy under both scoring procedures at every evaluated size from 143M to 803M. At 469M, the respective decreases are 1.11 and 0.54 pp under `acc`, and 1.32 and 2.61 pp under `acc_norm`. The 56M model is an exception: PIQA improves under both procedures, while the direction of the HellaSwag difference depends on the scoring procedure.

Filtering effects also vary across the remaining tasks and scoring procedures. CommonSenseQA improves `acc_norm` at every model size, whereas WinoGrande and MMLU show mixed effects. At 803M, the SciQ gain is 4.50 pp under `acc_norm` but only 0.20 pp under `acc`.The magnitude of the aggregate gain also depends on the task set. At 469M, HQ improves the four-task mean under `acc` by 2.11 pp, compared with 1.03 pp for the ten-task mean. Within the four-task aggregate, the gains on ARC-Easy and ARC-Challenge outweigh the losses on HellaSwag and PIQA. Thus, aggregate improvements should be interpreted alongside task-level results.

C.5 Performance Trajectories

To complement the final-checkpoint comparisons, Figure 5 shows the ten-task mean accuracy differences between HQ and FULL throughout training. HQ achieves higher mean accuracy at most evaluated checkpoints under both scoring procedures, indicating that the aggregate advantage is not confined to the end of training. Figure 6 shows the corresponding differences on ARC-Easy, ARC-Challenge, HellaSwag, and PIQA. The task-level trajectories reveal variation in both the magnitude

and direction of filtering effects over training, with ARC gains coexisting with HellaSwag or PIQA losses at multiple checkpoints. These results extend the final-checkpoint analysis by showing how task-level trade-offs evolve during training.

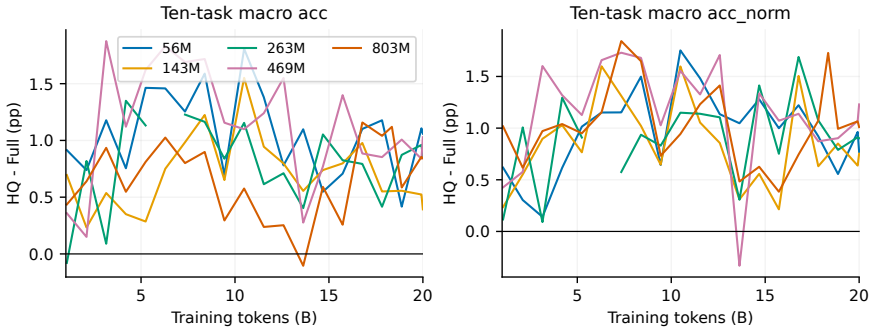

Figure 5: Ten-task mean accuracy differences between HQ and FULL over training.

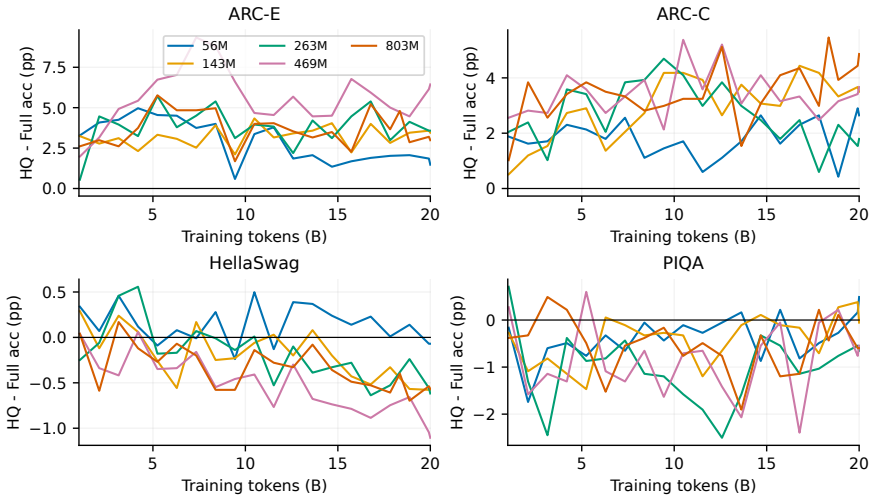

Figure 6: Task-level accuracy differences between HQ and FULL over training on ARC-Easy, ARC-Challenge, HellaSwag, and PIQA.

D LANGUAGE-ORDER SETTINGS AND TRAINING TRAJECTORIES

This appendix supplements the language-order analysis in the main text with experimental settings, the definition of the comparison metric, and additional results on language-specific performance during training.

D.1 EXPERIMENTAL SETUP

Language schedules. We compare three types of bilingual training schedules: sequence-level independent and identically distributed (IID) mixing with equal language probabilities, eight alternating monolingual phases, and two monolingual phases with a single switch in either direction. English-only and Chinese-only training provide additional reference conditions.

Models and training budgets. We train models with 43M and 143M parameters from random initialization and a sequence length of 1,024 tokens. The respective training budgets are 100M and 331M tokens, corresponding to approximately 2.37 training tokens per parameter at both sizes. For

each language, we compute negative log-likelihood on packed token blocks and divide it by the decoded UTF-8 byte count and $\ln 2$. We denote this byte-normalized packed-block loss by $\widetilde{L}$. To compare training schedules, we first express each language's loss as a relative change from its IID reference, then average the changes with equal weight:

$$P_s = \frac{100}{2} \sum_{\ell \in \{\mathrm{en,zh}\}} \left(\frac{\widetilde{L}_{\ell,s}}{\widetilde{L}_{\ell,\mathrm{IID}}} - 1 \right), \tag{3}$$

where s denotes the training schedule and ℓ denotes the evaluation language. The IID reference is taken at the same model size.

D.2 Effects of Language Ordering

Among the evaluated bilingual schedules, IID mixing achieves the lowest mean relative loss at both model sizes. Eight-phase alternation incurs smaller penalties than either single-switch schedule, but remains 4.57% above IID at 43M and 4.62% above IID at 143M(Figure 4b). Thus, dividing monolingual training into eight alternating phases improves performance relative to a single switch, while sequence-level interleaving performs best under the evaluated settings.

Single-switch training produces an asymmetric effect across languages. At 143M, training on English followed by Chinese increases English loss by 38.22% relative to IID, while Chinese loss increases by 1.25%. Reversing the order increases Chinese loss by 74.02%, while English loss is 1.21% lower than under IID. In both directions, the larger loss increase occurs in the language encountered first, indicating a recency effect in the final model's language-specific performance.

To characterize forgetting, we measure the increase in held-out loss from its minimum during training to its final value for each language. For English in the English-then-Chinese schedule, this increase is 0.31 at 43M and 0.46 at 143M. For Chinese in the Chinese-then-English schedule, the corresponding values are 0.83 and 1.10 (Figure 7). These within-language comparisons indicate more forgetting at 143M than at 43M.

E Language-Ratio Settings and Supplementary Results

This appendix supplements the language-allocation analysis in the main text with data-construction procedures, training and evaluation settings, and results for all five language ratios. The experiments examine how allocating a fixed training-token budget between English and Chinese affects held-out perplexity.

E.1 Experimental Setup

We train five 143M-parameter models with Chinese token shares $r \in \{0, 0.25, 0.50, 0.75, 1\}$, where $r = 0$ and $r = 1$ serve as English-only and Chinese-only reference models, respectively. Each optimizer update contains 64 blocks of 1,024 tokens, with exactly $64r$ Chinese blocks and $64(1-r)$ English blocks. Block positions are shuffled within each batch. Within each language, all conditions draw from a shared permutation of training blocks; conditions with smaller allocations consume prefixes of the blocks used by conditions with larger allocations, without repeating blocks. This construction controls the ordering of available data within each language while varying its training allocation. All models use the Haidass tokenizer and the same test sets, comprising 5,658 English and 4,967 Chinese documents.

E.2 Data Construction and Evaluation Protocol

We construct English and Chinese pools from `openbmb/Ultra-FineWeb`, retaining its upstream filtering without additional quality-score selection or synthetic data. Within each language, shards are deterministically shuffled and read in their original row order until the training pool contains at least 1B tokens, retaining the final document in full. All models use the Haidass-143M architecture and the shared initialization and language-mixing procedure described above. Training uses MindSpeed-LLM v2.3.0 on one Ascend 910B3 per run, with 15,258 optimizer updates, a microbatch size of 32, and two gradient-accumulation steps .

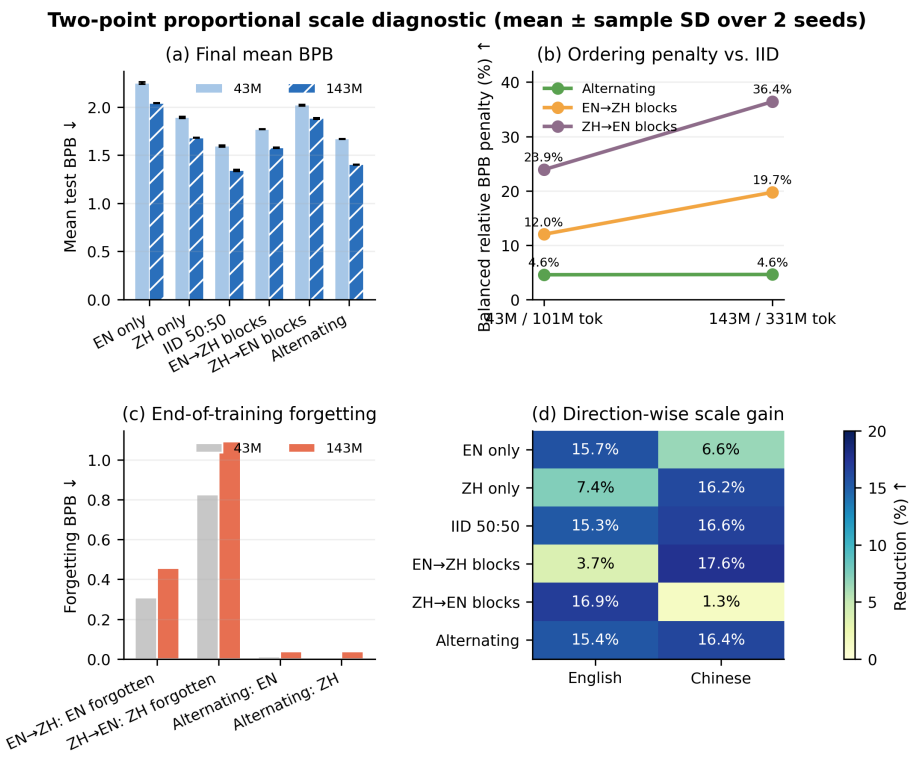

Figure 7: Supplementary comparison of language-order schedules at 43M and 143M.

We use Adam with $(\beta_1, \beta_2) = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-5} on all parameters, and gradient clipping at 2.0. The learning rate reaches 1.5×10^{-3} after 102 warm-up updates and then follows a cosine decay to zero.

For language ℓ and Chinese token share r, we compute

$$\begin{aligned} \mathrm{PPL}_{\ell,r} &= \exp\left(\frac{\sum_d N_{d,\ell,r}}{\sum_d T_{d,\ell}}\right), \\ \Delta_{\ell,r} &= 100\left(\frac{\mathrm{PPL}_{\ell,r}}{\mathrm{PPL}_{\ell,\mathrm{mono}}} - 1\right), \end{aligned} \tag{4}$$

where $N_{d,\ell,r}$ is document negative log-likelihood in natural-log units and $T_{d,\ell}$ is the number of scored tokens. The monolingual reference uses $r = 0$ for English and $r = 1$ for Chinese. Data-construction scripts, tokenizer files, evaluation outputs, and the source-version manifest are available in `umeiko/haidass-eval`.

E.3 Results and Interpretation

Table 10 reports results for all five ratios, and Figure 4(a) summarizes the relative changes for the bilingual mixtures. Across the tested ratios, increasing the Chinese token share decreases Chinese perplexity and increases English perplexity. At a 50:50 allocation, English and Chinese perplexities are 25.69 and 28.65, compared with their respective monolingual references of 21.90 and 23.92. The relative increases are 17.32% for English and 19.75% for Chinese, with paired 95% document-bootstrap intervals of [16.96, 17.74]% and [18.36, 20.86]%, respectively.

Asymmetric allocations shift this trade-off. At 25% Chinese, the relative increases over the monolingual references are 7% for English and 51.37% for Chinese. At 75% Chinese, they are 38.58% and 5.73%, respectively. Allocating more of the fixed budget to a language therefore reduces its perplexity penalty under the evaluated conditions.

Table 10: Language-ratio results for 143M models trained on approximately 1B tokens.

ZH share	EN PPL [95% CI]	ZH PPL [95% CI]	EN Δ (%)	ZH Δ (%)
0%	21.90 [21.25, 22.54]	3426.10 [3182.92, 3714.58]	0.00	+14221.62
25%	23.43 [22.73, 24.13]	36.21 [34.92, 37.61]	+7.00	+51.37
50%	25.69 [24.93, 26.45]	28.65 [27.56, 29.74]	+17.32	+19.75
75%	30.35 [29.45, 31.23]	25.29 [24.35, 26.26]	+38.58	+5.73
100%	170.76 [166.52, 174.85]	23.92 [23.04, 24.90]	+679.75	0.00

These results characterize the language-allocation trade-off under a fixed 1B-token budget and a single initialization. Because changing the ratio also changes each language's training exposure, the PPL differences from monolingual references cannot be attributed solely to cross-lingual interference. The preferred allocation for longer training remains to be established.

F Translation Adaptation: Training and Evaluation

This appendix supplements the translation-adaptation experiments in the main text with details of the three-stage training procedure, decoding settings, and evaluation protocol. Haidass-Translate-143M uses the same model weights for greedy decoding and direction-specific beam search. The suffix "beam" denotes a decoding configuration, not a separately trained model.

Stage 1: supervised fine-tuning. We initialize training from Haidass-143M and use 7.84M cleaned Chinese–English parallel sentence pairs. Each pair is converted into examples for both translation directions, yielding approximately 15.67M training examples. This stage uses only translation data. Training uses 16 Ascend 910C devices and a global batch size of 256. The learning-rate schedule is cosine decay with a peak of 3×10^{-5}, defined over five epochs (11,867 optimizer updates, approximately 6.2B tokens). The checkpoint selected using FLORES-200 dev occurs at 4.50 epochs and initializes Stage 2.

Stage 2: annealing on a curated data mixture. We continue training on 715,658 curated sentence pairs, with a cosine learning-rate schedule decreasing from 3×10^{-6} to 3×10^{-7}. The mixture excludes web-mined corpora with alignment noise or substantial amounts of traditional Chinese or Cantonese text. Chinese translations in TED and OpenSubtitles are regenerated using Qwen3.8-27B. The mixture also includes 48,000 BWB web-novel pairs not used in Stage 1, together with data from CoVoST2, BSTC, News, Wikimedia, and UN corpora.

Stage 3: sequence-level distillation. The final stage uses Qwen3.8-27B as the teacher, combining 70% teacher-generated targets with 30% original corpus data. The training set contains 1,579,486 examples, packed into 60,706 sequences. Training proceeds for 119 optimizer updates (approximately 0.5 epoch) with a global batch size of 256 and a cosine learning-rate schedule from 2×10^{-6} to 2×10^{-7}.

Decoding and model selection. We evaluate Haidass-Translate-143M with greedy decoding and direction-specific beam search. Greedy decoding uses `do_sample=false`, `num_beams=1`, and `repetition_penalty=1.0`. Beam search uses three beams with a length penalty of 1.25 for English-to-Chinese translation and four beams with a length penalty of 0.9 for Chinese-to-English translation. Both beam configurations use a maximum generation length of 256 new tokens.

FLORES-200 dev, containing 997 sentences per direction, is used for checkpoint and decoding-configuration selection. After selection, the model and decoding configurations are evaluated once on FLORES+ devtest. Table 4 reports both sets of results.

Evaluation metrics. We compute corpus BLEU using sacreBLEU with `tokenize=zh` for Chinese targets and `tokenize=13a` for English targets. We also report chrF++ with `word_order=2`. For Haidass-Translate-143M, evaluation prompts use the same chat-template formatting as training.

Training–evaluation overlap analysis. We examine overlap between 15.83M training examples and the FLORES-200 dev and FLORES+ devtest sets. Sentences are divided into consecutive fragments of eight English words or ten Chinese characters. A training example is counted as a match if it shares at least one fragment with a sentence in the corresponding evaluation set. The analysis identifies 1,147 matching training examples (0.0072%) for FLORES-200 dev and 1,788 (0.0113%) for FLORES+ devtest. Manual inspection attributes the detected matches to common phrases rather than full-sentence overlap.These findings characterize overlap under the specified fragment-matching procedure; they do not exclude other forms of overlap.