upstage

Solar Open Technical Report

Upstage Solar Team

Date: Jan 5, 2026
Website: https://upstage.ai

Abstract

We introduce Solar Open, a 102B-parameter bilingual Mixture-of-Experts language model for underserved languages. Solar Open demonstrates a systematic methodology for building competitive LLMs by addressing three interconnected challenges. First, to train effectively despite data scarcity for underserved languages, we synthesize 4.5T tokens of high-quality, domain-specific, and RL-oriented data. Second, we coordinate this data through a progressive curriculum jointly optimizing composition, quality thresholds, and domain coverage across 20 trillion tokens. Third, to enable reasoning capabilities through scalable RL, we apply our proposed framework *SnapPO* for efficient optimization. Across benchmarks in English and Korean, Solar Open achieves competitive performance, demonstrating the effectiveness of this methodology for underserved language AI development.

1 Introduction

1.1 Motivation

The open model ecosystem is pivotal for democratizing access to Large Language Models (LLMs), fostering transparency and community-driven innovation. In reality, however, the democratization is far from complete, failing most of the world's languages. While multilingual initiatives like Aya (Üstün et al., 2024) and Pangea (Yue et al., 2024) demonstrate progress, the landscape remains dominated by English and Chinese – the only two languages with both mature data availability and multiple frontier open models. Leading open models reflect this asymmetry: Qwen (Yang et al., 2025), DeepSeek (Liu et al., 2024), and Kimi (Team et al., 2025) prioritize English and Chinese; OLMo (OLMo et al., 2024) focuses solely on English. For other languages, neither large-scale datasets nor frontier models exist in comparable quantity or quality.

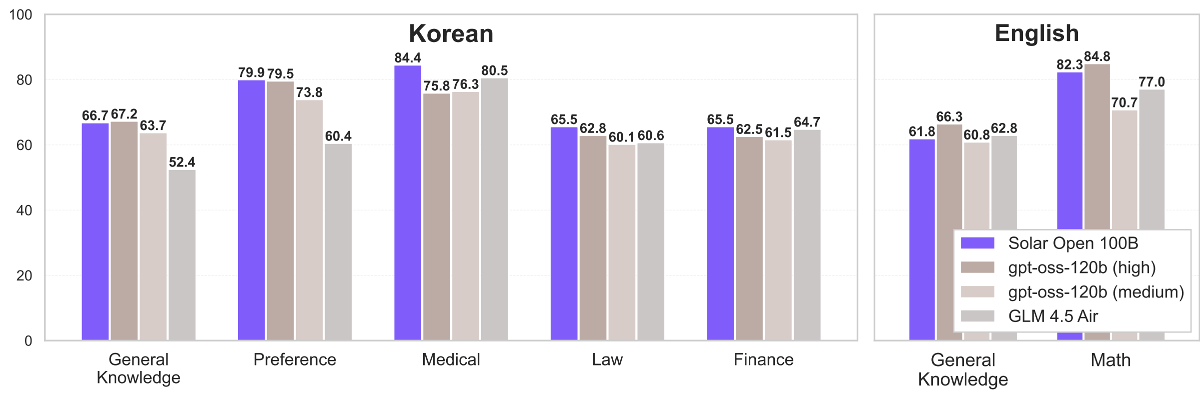

Figure 1: Overall performance of Solar Open and other comparable models.

Without language-specific considerations, however, models suffer degraded performance in downstream tasks. Language is intrinsically linked to its speakers and cultural context (Naous et al., 2024; Myung et al., 2024), fundamentally reshaping knowledge and task definitions (Son et al., 2025; Jang et al., 2022). Models also suffer from suboptimal tokenization, where reliance on byte-level fallback inflates sequence lengths and dilutes semantic density (Petrov et al., 2023; Rust et al., 2021).

These structural inefficiencies and cultural blind spots disproportionately impact underserved languages, driving language-specific model development efforts. We focus on Korean, alongside parallel efforts for Japanese (LLM-JP, Aizawa et al. (2024)), Arabic (Jais, Sengupta et al. (2023)), and other Korean initiatives (KORMo, Kim et al. (2025b)). Korean occupies merely 0.8% of indexed web content and ranks 17th in FineWeb 2 by byte count, facing reasonably severe data scarcity.

To address these challenges, we present Solar Open, Upstage's flagship open-weight LLM trained on 20 trillion tokens. Solar Open is built on a Mixture-of-Experts (MoE) architecture with 102 billion total parameters and 12 billion active parameters per token (Shazeer et al., 2017). The model is designed for both Korean-centric capabilities and general-purpose reasoning, addressing the data scarcity and reasoning challenges outlined above through a systematic methodology detailed in the following sections.

1.2 Challenges and Our Solutions

In this section, we present three major challenges we face when building Solar Open. At a high level, these challenges are related to bilingual data and reasoning. While their interconnections are revisited at the end of the section, we first describe our understanding of reasoning capabilities that motivates our data strategy.

Our approach is grounded in recent progress of how LLMs develop reasoning capabilities across training stages. During pre and mid-training, models encounter countless individual logical steps scattered across diverse documents (Ishibashi et al., 2025; Zhang et al., 2025). During supervised fine-tuning (SFT), they learn complete reasoning trajectories towards success – sequences of these logical steps that successfully solve problems, where the structure of reasoning matters more than specific content (Ruis et al., 2024; Li et al., 2025). Finally, during reinforcement learning (RL), models learn to compose and recombine these learned steps (Cheng et al., 2025), potentially forming novel reasoning paths not directly observed in training data (Han et al., 2025). This understanding motivates our strategies: we focus on diversity when synthesizing logical step demonstrations during pre-training (especially mid-training), curate successful multi-step trajectories for SFT, and use RL to teach compositional reasoning over these building blocks.

A. Synthetic Data for K-Data Scarcity and RL Besides English and Chinese, most languages face data scarcity for training LLMs. Occupying only 0.8% of the indexed web and ranking 17th by byte count in FineWeb 2, Korean exemplifies such an issue – its text content lacks in both quantity and quality (Penedo et al., 2025).[1] Thus, collecting data is far from enough. Training a strong Korean LLM, or LLMs targeting any other languages beyond English and Chinese, requires *generating* data that covers the entire spectrum of domains, quality tiers, and formats for training stages, matching the readily available English data.

Synthetic data plays a dual role in our approach: addressing Korean language scarcity while simultaneously creating RL-oriented training data aligned with our reasoning framework. We generate data aggressively – 4.5T tokens of high-quality, domain-specific synthetic data for pre-training through diverse augmentation, filtering, and transformation pipelines of data sources and ML models (Section 3). As discussed earlier in this section, this synthesis extends across

[1] The indexed web statistics is from `https://w3techs.com/technologies/overview/content_language`.

training stages with distinct purposes: (a) mid-training generates queries paired with diverse reasoning trajectories to enrich atomic logical operations, (b) SFT curates successful trajectories demonstrating effective problem-solving, and (c) RL employs generated queries to train compositional reasoning. This approach systematically targets each capability and language – reasoning, safety, agent workflows, and cultural alignment.

B. Bilingual and Reasoning-Targeted Curriculum Pre-training a 102B-parameter model requires a sophisticated data curriculum, and bilingual optimization introduces compounding complexity: Korean and English – the target language and lingua franca – must be balanced across training stages, quality thresholds must account for language-specific characteristics, and domain coverage must be maintained in both languages. This pattern applies beyond Korean to any underserved language paired with English.

Rooted in the same motivation with the data generation, our curriculum coordinates and optimizes the synthetic data across training stages, serving dual purposes: balancing bilingual data while supporting our reasoning framework's progression from atomic logical steps to complex reasoning trajectories. We implement a multi-phase curriculum (Section 3) with general quality classification, educational quality scoring with language-aware thresholds, and embedding-based topic clustering to ensure balanced domain coverage. Early phases expose the model to broad, diverse logical steps; later phases refine capabilities with high-quality data emphasizing Korean cultural knowledge and advanced reasoning. This curriculum, combined with aggressive synthetic data generation, enables the Solar Open base model to achieve strong performance at a lower token budget, demonstrating the efficiency of our approach (more details at Figure 5).

C. Scalable RL Framework Reinforcement learning enables the compositional reasoning that our framework targets (Cheng et al., 2025; Han et al., 2025). However, scaling RL for diverse objectives presents significant challenges. Traditional online RL tightly couples data generation, reward computation, and training – when targeting multiple capabilities simultaneously (reasoning, safety, preference alignment, cultural nuances), this coupling requires expensive infrastructure retuning for each objective, limiting scalability.

We address this through SnapPO (Section 6), a cyclic off-policy framework that decouples these three stages into independent processes with cached intermediate results. This decoupling enables two critical benefits: (1) linear scaling, where adding compute nodes directly increases throughput without infrastructure redesign or synchronization bottlenecks from stragglers, and (2) flexible multi-domain composition, where rewards are computed independently per domain and merged during training.

We leverage SnapPO to structure RL training across multiple objectives – reasoning optimization and alignment – where each composes data from specialized domains with targeted rewards. Similar decoupled approaches have been explored in frameworks like PRIME-RL (Intellect, 2025).

Interconnections These three challenges are deeply interconnected, unified by a curriculum-coordinated approach to bilingual data and reasoning development. Data scarcity drives aggressive synthetic generation (4.5T tokens), which the progressive curriculum coordinates across stages – balancing quality, domains, and languages while simultaneously preparing for RL through reasoning trajectory synthesis. This RL-oriented data strategy feeds directly into SnapPO, whose decoupled architecture makes multi-domain, multi-objective training tractable at scale. The result is a cohesive methodology where data generation, curriculum design, and RL framework mutually reinforce to overcome the challenges facing underserved languages. While we demonstrate this through Korean, the approach applies to any language lacking sufficient training data and frontier-quality models.

2 Model Architecture

There are two major aspects to a model architecture of any LLM; a tokenizer for preprocessing text and a network architecture for processing input text. In this section, we first describe our tokenization approach that ensures Korean is well represented, and move on to describing our choice of a network architecture focusing on its mixture-of-expert nature.

2.1 Solar Open Tokenizer

The design of Solar Open Tokenizer is optimized to achieve the goal of Solar Open to model Korean text and perform reasoning tasks. It is a custom-built byte-level Byte-Pair Encoding (BPE) tokenizer (Sennrich et al., 2016), specifically engineered with a large vocabulary size of 196,608 and trained on a large corpus that oversamples Korean and the target domains, as shown in Table 1.

Table 1: The data composition for training Solar Open Tokenizer.

Topic	Data Sources	Proportion
English	Refined web corpus, Wikipedia, arXiv	40%
Korean	High-quality Korean web text	22%
Code	GitHub, StackOverflow, etc.	12%
Math	Math StackExchange, etc.	10%
Multilingual	Japanese, Chinese, European languages	8%
Domain Specific	Finance, Legal, Medical documents	4%

2.1.1 Algorithm and Normalization

The tokenizer was trained using the Hugging Face `tokenizers` library with a BPE trainer. We implemented specific pre-tokenization rules using Regex to enhance performance in reasoning and code generation:

- **Digit Splitting:** We enforced a rule to treat all digits as individual tokens (using the pattern `\p{N}` in isolation). This design choice prevents the fragmentation of numbers into arbitrary subwords, significantly improving the model's arithmetic reasoning capabilities and its ability to parse scientific formulas by ensuring that numbers are represented consistently.
- **Whitespace Preservation:** Our regex-based pre-tokenizer preserves whitespace patterns, which is critical for programming languages that rely on indentation (e.g., Python). This ensures high fidelity in code generation tasks.

2.1.2 Chat Template

The Solar Open chat template adopts a structured message protocol design that balances compatibility with modern API patterns while optimizing for training stability and reasoning controllability. The template supports four role types (system, user, assistant, and tool) with native support for parallel tool calling, essential for agentic workflows.

A key design choice is the explicit separation of reasoning paths using the `<|think|>` token, allowing the model to generate internal chain-of-thought reasoning before producing final responses. This separation provides two critical benefits: (1) it facilitates more precise reward modeling during reinforcement learning by isolating reasoning from final outputs, and (2) it enables efficient reasoning path management in multi-turn conversations, where intermediate reasoning can be selectively retained or removed without affecting conversation history.

2.1.3 Evaluation on Tokenizer Efficiency

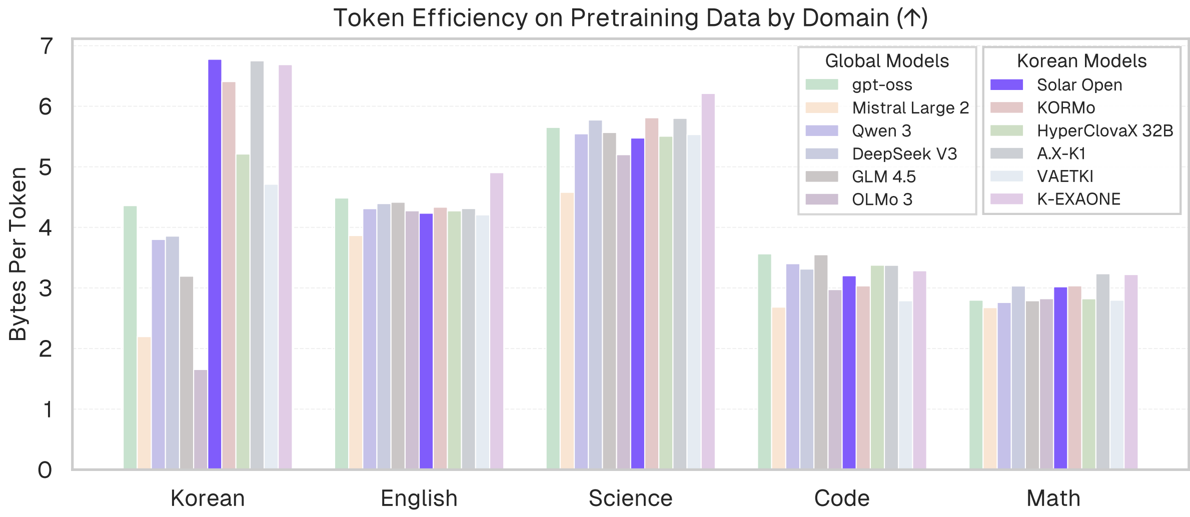

Figure 2: The compression rates of the Solar Open Tokenizer and other tokenizers (higher is more efficient). In each bar group, the nine bars left to Solar Open represent the 'global' (English and/or Chinese-centric) models, and the four bars right to Solar Open represent the Korean-centric models.

Preliminary evaluations show that the Solar Open Tokenizer achieves competitive compression rates measured by Bytes per Token, as in Figure 2. The baselines consist of various open models such as gpt-oss (Agarwal et al., 2025), Mistral Large 2 (Team et al., 2024), Qwen 3 (Yang et al., 2025), DeepSeek V3 (Liu et al., 2024), GLM 4.5 (Zeng et al., 2025), OLMo 3 (Olmo et al., 2025), as well as several Korean-specific open models such as KORMo (Kim et al., 2025b), HyperCLOVAX-SEED-Think-32B (Team, 2025b) A.X K1 (SK Telecom, 2025), VAETKI (Consortium, 2025), and K-EXAONE (LG AI Research, 2025). The benchmark is done on 10,000 samples of various open source datasets available on Hugging Face hub.[2] Note that compression rate is inherently tied to vocabulary size, which represents a fundamental design choice in tokenizer development. For a 102B-parameter model like Solar Open, allocating a larger vocabulary (196,608 tokens) optimizes the trade-off between compression efficiency and inference speed, as embedding table costs remain negligible relative to total model parameters. The training method follows tokka-bench (Gubler, 2025) Detailed vocabulary size comparisons across models are provided in Appendix A.2.

The first group, Korean, demonstrates the importance of oversampling target languages. The significantly lower compression rate of the "global" models indicates that without a careful design, oversampling, and training, the model would suffer from inefficient text encoding in various aspects – the training throughput, inference speed, and effective context length. Furthermore, the byte fallback would subsequently result in sub-optimal semantic segmentation, degrading the performance of the model. Conversely, all the Korean models outperformed the global models on Korean data text, with Solar Open, A.X-K1, and K-EXAONE achieve best performance together, closely followed by KORMo.

In other target domains – English, Science, Code, and Math – the deviations among models are relatively small, with Solar Open achieving competitive compression rates. This is due to long-tail distributions of token frequencies, where the most frequent tokens are already well-represented in any tokenizers.

[2] `maywell/korean_textbooks` (Korean Wikipedia), `Gwanwoo/cleaned_english_wiki` (English Wikipedia), `math-ai/AutoMathText` (Math), `Nbardy/science-theory-textbooks` (Science), `parameterlab/scaling_mia_the_pile_00_Github` (Code).

In addition to training-time efficiency, tokenizer efficiency also becomes critical at inference, where models are repeatedly invoked in real-world deployments. Unlike pretraining corpora, inference-time text typically consists of well-structured, concise outputs with distinct distributions, especially when explicit reasoning traces are present. As such, tokenizer efficiency at inference should be evaluated directly on actual model outputs rather than on raw web-scale text.

To this end, we measure inference-time tokenization efficiency using bilingual instruction-tuning outputs drawn from the IF-bilingual-sft dataset,[3] covering four settings: Korean non-reasoning, Korean reasoning, English non-reasoning, and English reasoning. For each setting, we sample 10,000 instruction–response pairs and compute Bytes per Token over the generated outputs, including explicit reasoning segments. Figure 3 summarizes the results. Overall, the inference-time results highlight a clear advantage of the Solar Open tokenizer, particularly in Korean settings. In Korean non-reasoning outputs, Solar Open achieves 4.69 Bytes per Token, outperforming gpt-oss (3.45, +36%) and DeepSeek V3 (3.19, +47%), and maintaining a noticeable margin even over strong Korean-specialized models such as KORMo (4.47, +5%). This trend continues in Korean reasoning outputs, where Solar Open reaches 4.83 Bytes per Token, exceeding gpt-oss (3.61, 34%) and remaining consistently higher than most baselines. Solar Open also demonstrates competitive performance on English, especially on non-reasoning.

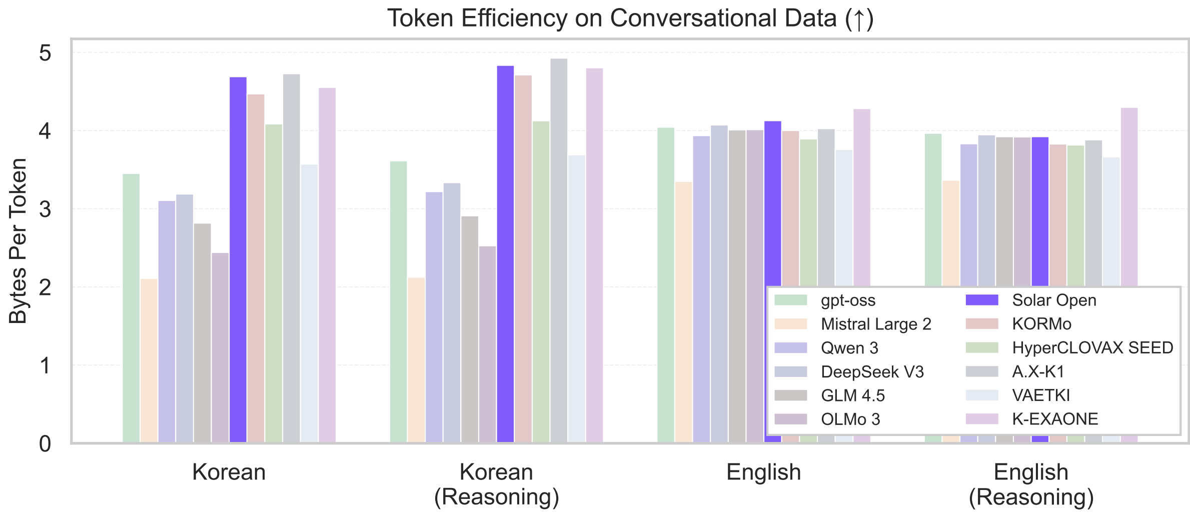

Figure 3: Inference-time tokenizer efficiency across languages and reasoning settings.

2.2 Mixture-of-Experts Architecture

Solar Open employs a sparse Mixture-of-Experts (MoE) Transformer architecture (Shazeer et al., 2017; Vaswani et al., 2017), chosen for its superior training and inference efficiency. Our design prioritizes efficiency within resource constraints while minimizing risk through proven architectural patterns. Given our computational resources of 480 B200 GPUs and a target to complete pre-training on 20 trillion tokens within a three-month timeline, we determine that a model with approximately 100 billion total parameters and 10 billion active parameters represents both a feasible and effective configuration. We also anticipate future up-scaling of Solar Open to about a 200B model, based on Depth-Up Scaling (Kim et al., 2024a).

[3] `https://huggingface.co/datasets/KORMo-Team/IF-bilingual-sft`, Kim et al. (2025b)

Table 2: Solar Open Architecture Specifications

Hyperparameter	Value
Total Parameters	102.6B
Active Parameters per Token	12B
Context Length	131,072
Vocabulary Size	196,608
Num Layers	48
Hidden Size	4,096
Intermediate Size	10,240
MoE Intermediate Size	1,280
Num Attention Heads	64
Num Key-Value Heads	8
Head Dimension	128
Num Dense Layers	0
Num Experts (Total)	129
Num Experts (Routed)	128
Num Shared Experts	1
Num Experts per Token (Top-k)	8
Activation Function	SiLU (Elfwing et al., 2018)
Positional Embedding	RoPE (θ: 10^6) (Su et al., 2024)

Architectural Design Process We benchmark detailed hyperparameters against contemporary MoE architectures with similar sizes: gpt-oss-120b (Agarwal et al., 2025), Qwen3-235B-A22B (Yang et al., 2025), GLM-4.5 (Zeng et al., 2025), and DeepSeek-V3 (Liu et al., 2024).

Expert configuration undergoes validation through ablation studies on a 10B-A1B prototype evaluated with MMLU and HellaSwag benchmarks (Wang et al., 2024; Zellers et al., 2019). Testing shared expert configurations reveals that incorporating a single shared expert improves performance while maintaining equivalent throughput. This shared pathway assists when routing paths are uncertain or when specific tokens struggle to identify appropriate specialized experts. While reducing the number of activated experts (Top-k) and MoE dimensionality would improve efficiency, excessive reduction risks performance degradation. We therefore adopt configurations that remain close to the proven designs of gpt-oss-120b and GLM-4.5: routing that selects 8 from 128 experts plus one shared expert.

Expert imbalance represents a persistent challenge in MoE architectures, requiring careful mitigation to ensure stable training. While recent work explores auxiliary loss-free approaches using expert bias alone, examination of DeepSeek-V3 and GLM-4.5 reveals that these systems employ load balancing loss in conjunction with expert bias. We validate this combined approach on our 10B-A1B prototype. Experiments with expert bias alone (coefficients: 1e-2, 1e-3, 1e-4) demonstrate that while later layers achieve balance, early layers consistently exhibit severe imbalance. In our experiment, introducing load balancing loss resolves this issue effectively. Unlike GLM-4.5 and DeepSeek-V3, which employ dense initial layers for stability, we follow gpt-oss-120b in using MoE layers throughout, finding that expert bias (coefficient: 1e-3) combined with sequence-wise load balancing loss (coefficient: 1e-4) provides sufficient stability without requiring dense layers. This design choice prioritizes architectural simplicity while maintaining training stability through careful loss balancing.

Training stability guides our remaining architectural decisions through comparative experiments between our 10B-A1B MoE prototype and a 3B dense baseline across 50B–100B training tokens. These ablations determine our adoption of block-masked attention and inform our capacity allocation strategy. We systematically vary hidden dimension, MoE FFN dimension, and layer count to balance depth versus width, arriving at our final configuration: 102B total parameters with 12B active parameters per token.

Distinctive Choices Comparing Solar Open with gpt-oss-120b, Qwen-3, GLM-4.5, and DeepSeek-V3, we observe that SiLU activation is the only component universally shared across these architectures. Our model distinguishes itself through several design choices: (1) complete absence of dense layers, prioritizing architectural simplicity over the additional stability mechanisms employed by reference models; (2) an intermediate size of 10,240; (3) an MoE intermediate size of 1,280, both optimized for our capacity allocation strategy; and (4) a vocabulary size of 196,608, substantially larger than conventional choices, reflecting our emphasis on tokenization efficiency for bilingual operation as detailed in Section 2.1. These choices collectively reflect a design philosophy that balances inference efficiency, training stability, and linguistic coverage.

3 Pre-training

Building a robust Korean-centric LLM requires confronting a fundamental challenge in the open model ecosystem: the scarcity of high-quality Korean training data compared to English. To address this gap, we prepare about **19.7 trillion tokens** for the pre-training stage with two key innovations: (1) a **low-to-high quality curriculum** that progressively transitions from broad, noisy data to highly curated content, and (2) **aggressive synthetic data augmentation** (reaching 64% in later phases) generated exclusively by open models, ensuring both data diversity and license compliance. This approach allows us to maximize knowledge acquisition early in training while refining capabilities with high-quality data in later phases.

3.1 Data Construction and Composition

Our data construction strategy balanced scale, quality, and Korean representation. The final 19.7T token corpus comprises: English General (13.0T), Math & Code (3.7T), Korean General (1.1T), Japanese (0.8T), Multilingual (0.8T), and Domain-Specific content in Finance, Medical, and Legal domains (0.4T).

We employ a multi-pronged construction approach:

- **License-Compatible Aggregation:** We use all available openly licensed datasets to establish a broad foundation.
- **Domain-Specific Curation:** For Korean and English, we manually curate 0.4T tokens of specialized content in finance, legal, and medical domains to enhance domain expertise.
- **PDF Parsing:** We developed a custom PDF parsing pipeline to extract 0.4T tokens from structured documents, preserving formatting and semantic structure critical for technical content.
- **Synthetic Data Generation:** We generated 4.5T tokens of synthetic data using Solar Pro 2 (Upstage, 2025) and other permissive open-source models.

3.2 Curriculum Learning Strategy

3.2.1 Low-to-High Quality Progression

We designed a multi-phase curriculum where both data quality and synthetic data ratio progressively increase. This 'noisy-to-high quality' approach allows the model to acquire broad world knowledge from diverse sources early, then refine its capabilities on carefully curated content:

- **Phase 1 (Initial Foundation):** We began with a diverse, noisy corpus where synthetic data comprised 10%. At this phase, we applied only basic general quality filtering to retain maximum coverage and linguistic diversity.
- **Phase 2 (Progressive Refinement):** This phase consisted of three sub-phases with increasing quality thresholds:
  - **Phase 2.A:** Synthetic ratio increased to 32%. We introduced our full three-method filtering framework (detailed below) at Level 1 threshold, removing the lowest-quality content.
  - **Phase 2.B:** Synthetic ratio 36%, filtering at Level 2 threshold, retaining approximately the top 50% of content by educational quality.
  - **Phase 2.C:** Synthetic ratio peaked at 64%, filtering at Level 3 threshold, retaining only the top 35% of educational content. This aggressive filtering ensured exposure to high-quality reasoning patterns and factual knowledge.

- **Phase 3 (Specialization):** The final 1.5T tokens consisted of highly curated data targeting Korean cultural knowledge, advanced mathematics, and structured code repositories. This phase included 0.5T of manually curated Korean content and 0.5T of repository-level code to develop coherent long-form generation capabilities.

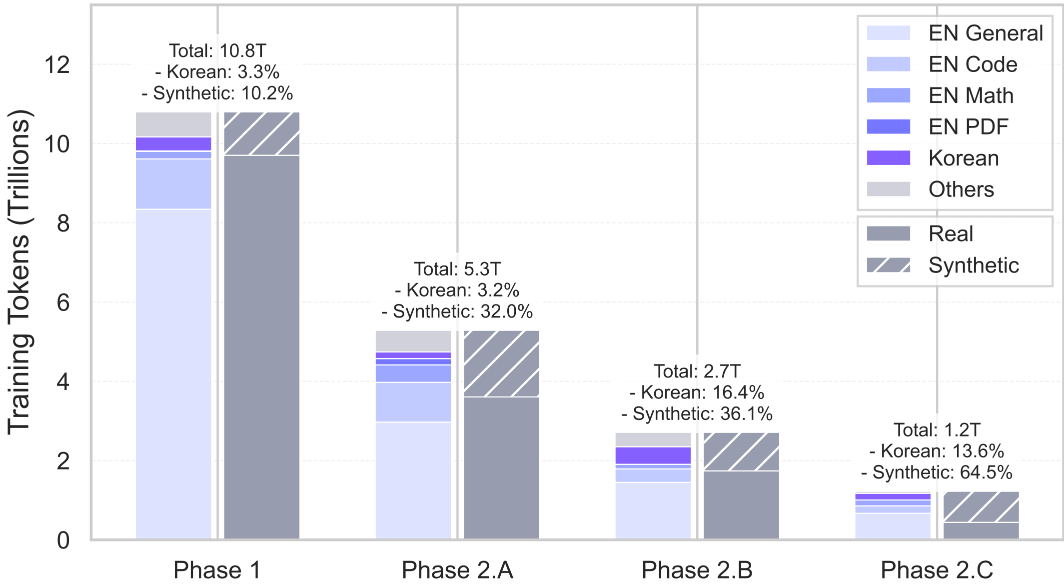

Figure 4: The data curriculum for the pre-training phases of Solar Open.

3.2.2 Multi-Layer Quality Filtering

To implement our progressive quality curriculum, we developed a three-method filtering framework that operates dynamically during data loading:

1. **General Quality Filtering:** A lightweight classifier trained to identify and remove noisy text (e.g., garbled encoding, excessive repetition, low coherence). This serves as a first-pass filter applied across all phases.

2. **Educational Quality Scoring:** Following FineWeb-Edu (Penedo et al., 2025), we train a regression-based model on human-annotated educational content to score each document's suitability for primary and grade school education (0–5 points). In later phases, we applied increasingly strict thresholds, eventually retaining only content scoring in the top educational percentiles (top 50% in Phase 2.B, top 35% in Phase 2.C).

3. **Embedding-Based Topic Filtering:** Following CLIMB (Diao et al., 2025), we used text embeddings to cluster the corpus and selectively sample from clusters aligned with the target knowledge domains such as science and technical documentation, as well as reasoning-focused content. This method ensured the coverage of critical topics even as overall data volume decreased.

These filters are integrated directly into our TorchTitan training pipeline (Liang et al., 2024), allowing dynamic application of filtering thresholds across phases without requiring multiple preprocessed dataset versions.

3.3 Engineering Optimization

Training Solar Open on 19.7 trillion tokens within a fixed timeline demands maximizing throughput on our 60-node Nvidia B200 cluster. Through systematic optimization across framework selection, communication patterns, MoE-specific kernels, and I/O pipelines, we increased training speed from 3,200 TPS on H200 to 4,000 TPS on B200, ultimately reaching 7,200 TPS – an 80% improvement over the B200 baseline. This section details our optimization approach and insights that may inform future large-scale MoE training efforts. Although subsequent framework updates have addressed some of these issues, we document our approach to inform similar large-scale deployments.[4]

3.3.1 Framework Selection

We evaluate multiple frameworks for MoE pre-training at 100-billion-parameter scale. DeepSpeed (Rasley et al., 2020) provides stability but relatively slow throughput compared to other frameworks. Megatron-LM (Shoeybi et al., 2019) shows high potential but its large and complex codebase presents significant implementation and debugging challenges, and exhibits unstable TPS for our specific MoE configuration. TorchTune (torchtune maintainers and contributors, 2024) delivers significant speedup over DeepSpeed (30–100% depending on configuration) but targets fine-tuning workloads rather than large-scale pre-training. We adopt TorchTitan (Liang et al., 2024), achieving over 50% TPS improvement and substantial memory savings compared to DeepSpeed. Our evaluation reveals two key insights: (1) for 100-billion-parameter scale models, larger batch sizes enabled by full activation checkpointing outweigh memory savings from selective checkpointing strategies; (2) expert parallelism and tensor parallelism provide no benefit over pure FSDP for our configuration, suggesting that architectural simplicity can outperform complex parallelization schemes at this scale.

3.3.2 Multi-Node Scaling Challenge

Standard FSDP2 (Zhao et al., 2023) performance degrades significantly when scaling from 16 to 60 nodes, with TPS dropping from 5,500 to 4,267. This degradation stems from all-reduce operations for gradient synchronization, which scale sub-linearly with node count due to inter-node communication volume. We address this by adopting Hybrid Sharding Data Parallel (HSDP), an extension of FSDP available in TorchTitan that divides the global device pool into smaller sharding groups. HSDP runs FSDP within each sharding group (10 nodes in our configuration) and synchronizes gradients across groups (6 replicas). This hierarchical structure confines most communication to intra-group operations while maintaining global gradient consistency through periodic inter-group all-reduce, achieving 26.5% throughput improvement and reaching 5,400 TPS at 60 nodes. This result demonstrates that exploiting the bandwidth differential between intra-node and inter-node communication effectively mitigates the saturation point of standard FSDP at large scale.

3.3.3 MoE-Specific Optimizations

Router Dtype Restoration. We identify a subtle implementation issue where the router's sigmoid operation correctly casts to FP32 for numerical stability but does not restore the original dtype afterward, causing subsequent matrix operations to execute in FP32. Restoring the dtype immediately after the sigmoid operation yields 13.7% speedup without affecting convergence. This demonstrates the importance of verifying dtype flow throughout computation graphs, particularly after precision conversions motivated by numerical stability.

[4] The pretraining is done as with a hybrid setup of FP8 and bfloat16, and we release Solar Open in bfloat16. This setup was chosen over a full FP8 training, considering the challenges in optimizing the throughput on the Nvidia B200 GPU as of 2025 August.

Load Balancing Loss Computation. Replacing one-hot encoding with histogram-based computation in the MoE load balancing loss reduces routing computation time by 20%. This optimization is critical during H200 tuning and applies effectively to B200 as well.

Expert Parallel Fast Path. TorchTitan's grouped GEMM implementation includes padding and reordering operations designed for expert parallelism configurations. When expert parallelism is disabled (as in our setup), these operations add unnecessary overhead. Implementing a conditional fast path bypass gains 14.5% TPS. This issue has since been addressed in upstream PyTorch, though the workaround is essential for early B200 deployment.

3.3.4 Hardware Adaptation for Early B200 Deployment

Deploying on B200 shortly after its release presents compatibility challenges with existing software stacks. Triton lacks CUDA 13.0 support, and the ScaledDotProductAttention backend prevents successful graph compilation. We manually patch Triton for compatibility and modify attention backends to enable compilation. Recent PyTorch updates resolve these infrastructure issues while adding FP8 support, delivering 11.3% speedup. We also encounter gradient norm instabilities when grouped GEMM operations process an excessive number of tokens simultaneously; limiting tokens per operation resolves this numerical issue and provides 5.8% additional speedup. Both the compilation compatibility and gradient norm issues have since been addressed in upstream PyTorch. These experiences underscore that early hardware adoption requires both technical workarounds and close coordination with framework maintainers.

3.3.5 Data Loading Optimization

Loading terabytes of pre-training data from VAST storage initially creates a significant bottleneck, requiring over 8 hours to initialize training. The slowest workers experience significantly longer delays due to I/O lock contention, making rapid iteration infeasible. Our solution exploits the internal structure of Arrow-formatted datasets: rather than all workers loading the complete dataset before sharding, each worker selectively loads only the Arrow files corresponding to its world rank partition. This file-level sharding approach reduces initialization time from 8 hours to approximately 8 minutes while maintaining flexibility for cluster size adjustments, unlike static pre-sharding strategies. This demonstrates that when I/O becomes the primary bottleneck, exploiting data format structure to enable embarrassingly parallel loading can yield substantial improvements.

Table 3: B200 throughput progression (4K sequence, 60 nodes). Percentages show gain over previous step.

Optimization	Gain	Cumulative TPS
B200 Baseline (from 3,200 TPS at H200)	–	4,000
Expert Parallel Fast Path	+14.5%	4,800
HSDP (60-node scaling)	+26.5%	5,400
Router Dtype Restoration	+13.7%	6,167
Repeat-KV Optimization	+3.3%	6,200
PyTorch Update (compile + FP8)	+11.3%	6,900
Grouped GEMM Token Limiting	+5.8%	7,200

Table 3 summarizes the cumulative optimization trajectory. Systematic attention to communication patterns, dtype handling, and hardware-specific issues nearly doubles training throughput from the B200 baseline.

3.4 Result

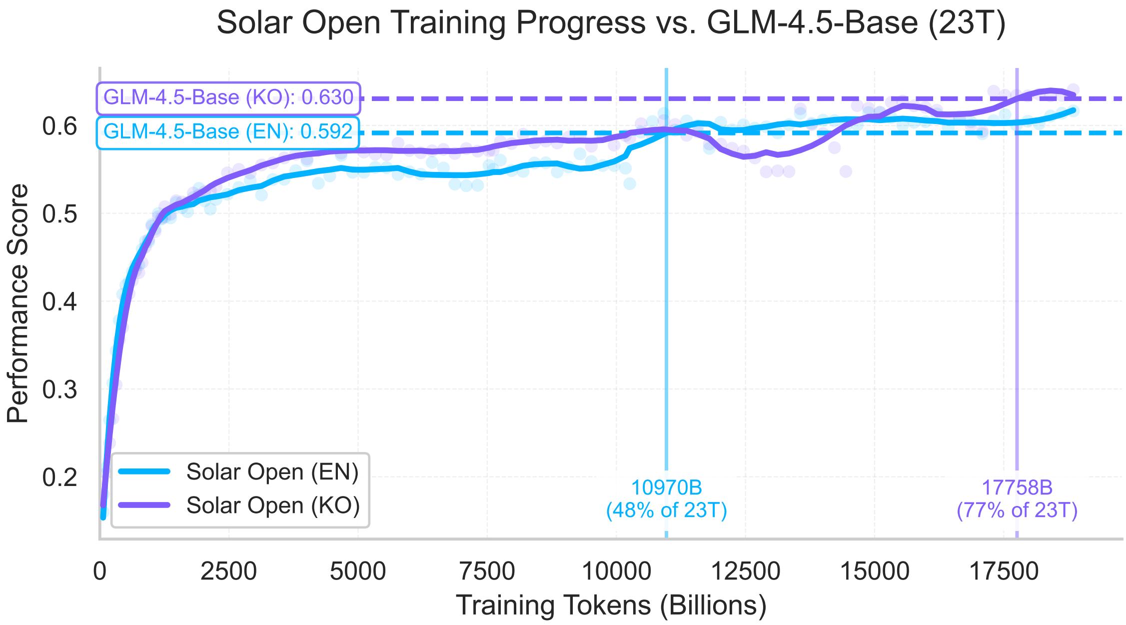

Figure 5: Training trajectory comparison between Solar Open and GLM-4.5-Base (23T tokens). Solar Open achieves comparable performance at 10.9T tokens (English) and 17.8T tokens (Korean), based on MMLU, MMLU-Pro, and HellaSwag benchmarks. Curves are smoothed for clarity.

To validate the efficiency of our data curriculum and synthetic data strategy, we conduct preliminary evaluations throughout pre-training by tracking performance on standard benchmarks. We compare Solar Open's training trajectory against GLM-4.5-Base (Zeng et al., 2025), a comparable-scale model (106B) trained on 23 trillion tokens. The evaluation employs MMLU (Hendrycks et al., 2020), MMLU-Pro (Wang et al., 2024), and HellaSwag (Zellers et al., 2019), measuring performance on both English and Korean versions of these benchmarks.

Figure 5 shows that Solar Open achieves GLM-4.5-Base's performance levels at approximately 10.7T tokens for English benchmarks and 17.8T tokens for Korean benchmarks – representing only 48% and 77% of GLM's total training budget, respectively. This substantial efficiency gain demonstrates the effectiveness of our bilingual curriculum design combined with aggressive synthetic data generation. The progressive refinement strategy, where synthetic data ratio increases from 10% to 64% while simultaneously raising quality filtering thresholds, enables the model to acquire competitive capabilities with significantly reduced token consumption.

While these results are preliminary and do not constitute a comprehensive evaluation, they provide empirical support for our core thesis: carefully designed data curriculum and synthesis strategies can dramatically improve training efficiency for underserved languages without compromising performance on English.

4 Mid-training

Following pre-training, we conduct a mid-training stage to enhance reasoning capabilities through RL-oriented data synthesis while preventing catastrophic forgetting. Our mid-training is similar to pre-training and shares the same engineering infrastructure, since the model continues to be trained on documents with the next-token prediction objective. This intermediate stage connects the broad knowledge acquisition of pre-training with the capability specialization of post-training.

RL-Oriented Reasoning Trajectory Synthesis A core component of our mid-training is RL-oriented reasoning trajectory synthesis. For challenging queries (identified by our difficulty estimator, see Section 5), we generate multiple diverse reasoning trajectories from open-source models. Rather than formatting these as chat conversations, we structure them as coherent documents containing the query followed by 2–5 different solution approaches. This format serves two purposes: (1) it provides the model with diverse logical step sequences during continued pre-training (Ishibashi et al., 2025; Zhang et al., 2025), enriching the repertoire of atomic reasoning operations (Cheng et al., 2025), and (2) it produces reasoning patterns that can be flexibly recombined during later RL stages.

This synthesis process generates approximately 50% of the reasoning category, which is 850B tokens and 64% of mid-training data. Solar Pro 2 and multiple permissive open-source models are used in this process. The diversity of source models and variation in problem-solving approaches ensures broad coverage of logical reasoning patterns.

Data Composition and Training Strategy The mid-training stage utilizes 1,150B tokens of specialized data in total, as shown in Table 4. Beyond reasoning trajectories, we include long-context data (135B tokens, 16%) to extend the model's effective window to 32k tokens and incorporate high-quality pre-training corpus from Phase 2.C (170B tokens, 20%) to prevent catastrophic forgetting. Korean and English each constitute approximately half of the data volume, while synthetic data comprises about 80% of the total. Within the reasoning category, general reasoning, code, and mathematics represent 37%, 16%, and 47%, respectively.

Table 4: Mid-training Data Composition. Among the reasoning category, synthetic data occupies about 50%.

Category	Tokens
Reasoning (64%)	850B
Long Context (16%)	135B
Subset of Phase 2.C (20%)	170B
Total	**1,150B**

5 Post-training: SFT

We focus on SFT (Ouyang et al., 2022), where our contribution lies in data construction and systematic evaluation using established training algorithms. Our novel RL methodology is discussed in Section 6. Training leverages the same TorchTitan infrastructure from pre-training (Section 3) with HSDP, compilation optimizations, and numerical precision management.

Our SFT data strategy builds directly on the reasoning framework established during pre-training and mid-training. While earlier stages provided the model with diverse logical steps and multiple solution paths, SFT curates successful, high-quality reasoning trajectories that demonstrate effective problem-solving (Li et al., 2025).

This stage is critical for mastering atomic reasoning skills (Cheng et al., 2025), a prerequisite for later RL-based composition. Beyond reasoning, SFT teaches instruction following, response formatting, and proper application of the chat template (Ouyang et al., 2022). We also incorporate successful trajectories discovered during preliminary RL runs, creating a feedback loop that continually refines the model's reasoning capabilities.

5.1 Difficulty-Aware Data Curation

Not all queries contribute equally to model capability development; training on queries that are too easy provides diminishing returns on already-mastered skills, while overly difficult queries provide insufficient learning signal. For SFT, we develop a difficulty estimator to enable strategic data allocation. This allows us to filter queries by difficulty threshold, apply enhanced quality control to challenging instances, and balance difficulty distribution across training data to ensure appropriate coverage of the capability spectrum.

We train a dedicated difficulty classifier based on self-consistency patterns across multiple capable models. The core insight is that difficult queries elicit divergent responses even from strong models, while easy queries produce consistent answers across models of varying capabilities. To implement this approach, we collect responses from five model configurations: GLM-4.5-Air (Zeng et al., 2025) with and without chain-of-thought reasoning, Qwen-3-30B (Yang et al., 2025) with and without chain-of-thought, and Solar Pro 2 (Upstage, 2025) without reasoning. For training data, we sample queries from several domains including math, code, science, and medical, and generate approximately 140K query-response tuples. We then construct pairwise comparisons and use LLM-based labeling to determine which query is more difficult in each pair. This labeled data trains our difficulty classifier.

The trained difficulty estimator guides SFT data curation:

- **Difficulty-Based Filtering:** We apply difficulty thresholds to curate training data appropriate for each stage, providing progressively challenging examples without overwhelming the model with queries beyond its current capability.
- **Enhanced Quality Control:** High-difficulty queries require stricter validation procedures, with manual inspection of reasoning paths and response quality before inclusion in training data.
- **Balanced Sampling:** We stratify data by difficulty to prevent over-representation of elementary queries while maintaining sufficient coverage across the difficulty spectrum.

This estimator also plays a critical role in RL training (Section 6), enabling difficulty-aware prompt selection and reward shaping.

5.2 Complex Query Generation

Raw query datasets exhibit unbalanced difficulty distributions, constraining both task coverage and reasoning capability development. We develop a query generator to scalably construct difficulty-balanced queries across diverse domains for SFT.

We define query generation as synthesizing relevant or semantically similar queries from seed contexts. We aggregate existing query data with related seed texts and fine-tune Solar Pro 2 (Upstage, 2025) as the generator. This approach leverages the base model's general capabilities to produce contextually appropriate queries from novel seeds. However, the model initially generates predominantly trivial queries ineffective for reasoning development. We address this through cyclic optimization: using the difficulty classifier from Section 5 to assess and stratify generated queries, we iteratively retrain the generator toward progressively higher complexity. By applying this optimized generator across diverse domain seeds, we construct a robust dataset of queries demanding reasoning capabilities across a wide spectrum of domains.

5.3 Agent Capability Development

Developing agentic capabilities requires training data that captures the full complexity of real-world tool use: decision-making, planning, tool selection, argument generation, result analysis, error handling, memory and context handling, and environment awareness. To address this, we construct two complementary agent simulation pipelines that synthesize diverse, multi-turn tool-use trajectories at scale.

Tool Synthesis and API Graph Construction Rather than relying solely on existing benchmark tools, we synthesize diverse tool sets by expanding API specifications from available training datasets. Instead of static API graphs, we utilize structured output format prediction and semantic expansion to model tool dependencies, enabling generation of realistic multi-step workflows. For each tool, we define parameters and expected return values through an analysis framework (domain, entities, and workflows), enabling trajectory generation that reflects real-world API usage patterns.

Task-Oriented Simulation The task-oriented simulator generates complex single-task scenarios requiring multi-step reasoning and tool use. Given a tool set, we synthesize task specifications with explicit success criteria and expected tool-use patterns. The simulator creates scenarios that require agents to decompose goals, select appropriate tools from the available set, generate correct arguments (including references to previous tool outputs), and validate results. We generate tasks across three complexity levels: easy (single-tool with constraints), medium (multi-tool with dependencies), and hard (extended multi-step with branching and error recovery). This pipeline produces 161,608 training samples spanning approximately 1.3 billion tokens.

User-Oriented Simulation In practice, users rarely express complex goals in a single utterance; instead, they decompose problems incrementally through dialogue. The user-oriented simulator models this interactive process by generating multi-turn conversations where a simulated user iteratively refines requirements, asks follow-up questions based on intermediate results, and adaptively guides the assistant. The simulator first generates a high-level task, then decomposes it into a sequence of sub-tasks presented across multiple turns. Each turn may depend on information from previous tool executions, requiring the agent to maintain context and synthesize partial results. This approach produces 177,375 training samples spanning approximately 3.0 billion tokens, with an average of 3.16 turns and 2.48 sub-tasks per conversation.

Training solely on this simulated data achieves 60 points on Tau^2-Bench (Barres et al., 2025) without dedicated RL training, demonstrating the effectiveness of high-quality simulation data for agentic capability development.

5.4 Korean Knowledge Integration

To enhance Korean cultural and historical knowledge, we construct specialized training data that is incorporated throughout our post-training pipeline (SFT, DPO, and RL). Korean cultural knowledge is typically fragmented across existing sources. We use embedding-based matching to construct multi-hop QA pairs that link related knowledge:

- **Comparative QA:** Pairs related documents for comparison or contrast reasoning.
- **Causal QA:** Chains multiple documents into cause-effect sequences.
- **Multi-Hop QA:** Builds on information from multiple sources through multi-turn dialogue.
- **Theme Inference QA:** Requires identifying common patterns or hidden intentions across diverse documents.

This multi-hop construction enables the model to develop relational understanding beyond memorization of isolated facts. In SFT and DPO, this knowledge-augmented data, through compositions, trains general Korean knowledge capabilities. During RL (Section 6), we extend this with targeted alignment for Korean cultural sensitivities and sensitive topics, incorporating culturally appropriate treatment of historical events, social issues, and cultural contexts using culturally-informed reward models.

5.5 Safety Framework

We develop a comprehensive safety framework addressing 38 risk categories. For each category, we define appropriate response strategies based on risk assessment. The categories span child safety, critical infrastructure, violence and hate speech, privacy violations, weapons and illegal goods, psychological harm, misinformation, political manipulation, surveillance, fraud, platform abuse, sexually explicit content, unregulated professional advice, emotional manipulation, and illegal activities.

- **Refuse with Redirection:** For requests that violate safety guidelines with clear harmful intent (e.g., bomb-making, hacking, fraud), the model explicitly refuses while providing educational context about the risks involved. See Section A.3.1 for examples.
- **Safe Completion:** For sensitive topics where outright refusal could itself be harmful – such as queries about self-harm or suicide – the model provides supportive, bounded responses that offer professional resources, crisis hotlines, and encouragement to seek help, without providing instructions that could facilitate harm. This strategy also applies to politically controversial topics and unregulated professional advice domains where providing balanced educational information is appropriate. See Section A.3.2 for examples.

To prevent over-refusal, we construct adversarial safety data – queries designed to appear unsafe but are contextually appropriate (e.g., educational discussions of historical atrocities, medical research questions). This is done by i) sampling adversarial safe query examples from the training data of WildguardMix (Han et al., 2024), and ii) creating their answers following our safety policy. Training on this data fine-tunes the model's safety responses, reducing false positives while maintaining robust protection against genuine risks. The distinction between refusal and safe completion is critical: for self-harm queries, providing empathetic support and professional resources is safer than simple refusal, which may prevent users from seeking necessary help.

6 Post-training: RL

Following SFT, we employ RL to optimize capabilities that require exploration and iterative refinement. While SFT establishes basic instruction-following and knowledge integration (Section 5), RL enables systematic capability optimization through reward-driven learning. We structure RL training in two sequential phases with distinct objectives: RL Phase A focuses on reasoning capability maximization, while RL Phase B optimizes human preference alignment and safety while maintaining reasoning performance.

A fundamental challenge in scaling RL is the tight coupling between data generation, reward computation, and training – changes to data distribution or reward functions require complete infrastructure reconfiguration. We address this through Snapshot Sampling for Policy Optimization (SnapPO), a cyclic off-policy framework that decouples these three steps, enabling independent optimization and rapid iteration. This architecture is particularly effective for testing and merging diverse domain-specific data with heterogeneous reward functions, enabling composition of complex training curricula from independently developed components.

6.1 SnapPO Framework

Our SnapPO framework operationalizes three-step decoupling through an iterative offline training loop. Each iteration consists of generation, reward computation, and training phases that execute independently with cached intermediate results.

Generation Step We employ vLLM (Kwon et al., 2023) for efficient response generation at scale. For each prompt in the training set, the current policy model generates multiple response candidates (typically 8-16 samples per prompt). Critically, we cache the behavior policy's log probabilities during generation, enabling off-policy learning in later steps. The generation step operates asynchronously from training, allowing us to pre-generate large response pools without blocking gradient updates.

Reward Computation Once responses are generated, we compute domain-specific rewards in a separate batch processing step. Different data types employ specialized reward functions: verifiable correctness for closed-ended STEM problems, multi-dimensional scoring for agent simulation, reward model-based evaluation for open-ended reasoning and writing, and pattern-based detection for degeneration issues. Reward computation is entirely decoupled from both generation and training, allowing us to iterate on reward design without retraining the policy. All rewards are cached to disk before training begins.

Training Step We employ Group Sequence Policy Optimization (GSPO) (Zheng et al., 2025), a variant of GRPO, for two strategic reasons. First, GSPO demonstrates superior stability when training sparse MoE architectures compared to conventional methods. Second, it obviates the need for an additional KL divergence term, significantly enhancing memory efficiency during training. This stage leverages the same TorchTitan infrastructure established during pre-training (Section 3), utilizing HSDP, compilation optimizations, and numerical precision management to ensure scalability.

Iterative Cycles Training proceeds through iterative cycles: in each cycle, the model is trained on a batch of prompts, followed by the updated policy generating new responses for the next prompt batch. This decoupled architecture provides several critical advantages: (1) independent tuning of generation throughput, reward complexity, and training hyperparameters, (2) rapid testing and composition of data from multiple domains with heterogeneous reward functions, (3) resource flexibility allowing specialized hardware allocation for each step, and (4) significantly reduced iteration cost when experimenting with new data distributions or reward designs.

6.1.1 SnapPO Implementation and Advantages

The decoupled architecture of SnapPO provides concrete engineering and efficiency benefits, enabling the compositional reasoning training that our framework targets (Cheng et al., 2025; Han et al., 2025).

Linear Scalability By separating generation from training, SnapPO achieves near-linear scaling with compute resources. Adding nodes increases throughput proportionally without requiring infrastructure redesign or hyperparameter re-tuning. This contrasts with coupled online RL approaches where generation and training compete for GPU memory and computation. Our implementation demonstrates consistent scaling across varying cluster sizes, maintaining efficiency as we expand to hundreds of GPUs.

TorchTitan Integration We implement SnapPO training using TorchTitan (Liang et al., 2024), which provides optimized FSDP and model parallelism. This integration delivers significantly faster training than alternatives such as verl (Sheng et al., 2024) in our benchmarks. The shared infrastructure between pre-training and RL training (HSDP, compilation optimizations) enables rapid transfer of engineering improvements across training steps.

Flexible Multi-Domain Composition The cached intermediate representation (generated responses and computed rewards) enables flexible mixing of data from diverse sources. Math RL, code RL, agent simulation, and safety data can be balanced dynamically without regenerating responses or recomputing rewards. This flexibility was essential for our two-phase RL approach covering both reasoning optimization and preference alignment. Similar decoupled approaches have been explored in frameworks like PRIME-RL (Intellect, 2025), demonstrating the broader applicability of generation-training separation in RL systems.

6.2 RL Phase A: Reasoning Optimization

The first RL phase focuses exclusively on maximizing reasoning capabilities across STEM domains, agent workflows, and complex problem-solving. We train on approximately 200K prompts sampled to emphasize challenging scenarios where exploration significantly improves over supervised learning. The prompts used for RL training originate from multiple sources developed in earlier stages. Mid-training query generation provides challenging STEM problems through difficulty-aware sampling, SFT agent simulation contributes diverse tool-use scenarios, and we synthesize additional open-ended reasoning queries. This prompt corpus reflects the capabilities targeted during RL training, with SnapPO enabling efficient multi-domain composition.

6.2.1 Data Composition

- **STEM Reasoning:** Closed-ended and open-ended mathematical and scientific problems with verifiable correctness rewards and process-based evaluation. Korean problems are entirely synthetic due to data scarcity and comprise approximately 50% of this category.
- **Code Generation:** Programming tasks with execution-based rewards and style/efficiency scoring.
- **Agent Simulation:** Multi-turn tool-use scenarios with composite rewards measuring task completion, interaction quality, and error recovery.

Prompts are filtered using the difficulty estimator from Section 5 to ensure appropriate challenge levels. We generate multiple response candidates per prompt, enabling the model to explore diverse reasoning strategies and consolidate effective approaches through reward-weighted gradient updates.

6.3 RL Phase B: DPO for Preference Alignment

The second RL phase shifts focus to human preference alignment while maintaining the reasoning capabilities established in Phase A. This phase addresses writing quality, safety, and degeneration handling. We adopt cyclic DPO for response alignment, where KL divergence regularization in the DPO loss prevents degradation of reasoning capabilities.

6.3.1 Data Composition

- **Human Preference:** Preference pairs covering STEM explanations, creative writing, and conversational quality, with model-based reward estimation.
- **Safety Alignment:** Scenarios spanning the 38 risk categories from Section 5, with dedicated rewards for appropriate refusal and safe completion strategies.
- **Degeneration Handling:** Detection and penalization of repetition, language errors, and formatting issues.
- **Korean Cultural Sensitivity:** Targeted alignment for culturally sensitive Korean topics, incorporating nuanced handling of historical events and social issues through specialized reward models.
- **Reasoning Maintenance:** A subset of Phase A data to prevent capability regression, ensuring preference optimization does not compromise reasoning performance.
- **Agent Data:** Scenarios for the model's self-correction behavior on agentic workflows.

This phase employs more conservative exploration compared to Phase A, prioritizing stable alignment over aggressive capability expansion. The inclusion of reasoning maintenance data ensures the model retains mathematical and agentic capabilities while optimizing for human preferences.

7 Evaluation

7.1 Tasks

We evaluate Solar Open across Korean and English benchmarks spanning general knowledge, domain expertise, reasoning, instruction-following, and preference alignment.

For Korean, general knowledge is assessed through KMMLU (Son et al., 2025), KMMLU-Pro (Hong et al., 2025), CLIcK (Kim et al., 2024b), HAE-RAE v1.1 (Son et al., 2024), and KoBALT (Shin et al., 2025). Domain-specific capabilities are measured using KBankMMLU (finance, derived from recent national and accredited private qualification exams), KBL (law) (Kim et al., 2024c), and KorMedMCQA (Kweon et al., 2024) (medical). Mathematical reasoning is evaluated on Ko-AIME 2024/2025 (translations of official AIME exams) and HRM8K (Ko et al., 2025). Instruction-following is tested with Ko-IFEval (Korean adaptation of IFEval), an in-house dataset used in (Kim et al., 2025a)), and preference alignment with Ko Arena Hard v2, another in-house dataset that translates the Arena Hard v2 (Li et al., 2024b).

For English, we use MMLU (Hendrycks et al., 2020) and MMLU-Pro (Wang et al., 2024) for general knowledge, GPQA-Diamond (Rein et al., 2024) for graduate-level science, and HLE (Phan et al., 2025) for high-level expertise. Mathematical reasoning is tested on AIME 2024/2025[5] and HMMT 2025 (Feb/Nov) (Balunović et al., 2025). Code generation is evaluated using LiveCodeBench v6 (Jain et al., 2024). Instruction-following is measured through IFBench (Pyatkin et al., 2025) and IFEval (Zhou et al., 2023), preference alignment via (Li et al., 2024a,b) and Writing Bench (Wu et al., 2025), agentic capabilities on Tau^2 (Airline/Telecom/Retail) (Barres et al., 2025), and long-context understanding through AA-LCR (Team, 2025a).

7.2 Result and Discussion

Solar Open demonstrates strong Korean capabilities while maintaining strong English performance (see Table 5 and Table 6). On Korean benchmarks, the model consistently outperforms gpt-oss-120b-medium across nearly all categories and surpasses gpt-oss-120b-high in several key areas. For general knowledge, it achieves 73.0 on KMMLU (+2.7pp over gpt-oss-high), 64.0 on KMMLU-Pro (+1.4pp), 78.9 on CLIcK (+1.7pp), and 73.3 on HAE-RAE v1.1 (+2.5pp). Domain-specific performance is particularly strong: 65.5 on KBankMMLU for finance (leading all baselines by +0.8–4.0pp), 65.5 on KBL for law (+2.7pp over gpt-oss-high), and 84.4 on KorMedMCQA for medical domains (+3.9pp over GLM-4.5-Air, +8.6pp over gpt-oss-high). The model achieves 79.9 on Ko-Arena Hard v2 for preference alignment, surpassing even gpt-oss-120b-high (+0.4pp) and substantially outperforming GLM-4.5-Air (+19.5pp). Mathematical reasoning shows competitive results at 80.3/80.0 on Ko-AIME 2024/2025 and 87.6 on HRM8K, while instruction-following achieves 87.5 on Ko-IFEval.

On English benchmarks, Solar Open delivers performance comparable to or exceeding gpt-oss-120b-medium across most categories. General knowledge scores are competitive: 88.2 on MMLU (+0.3pp vs medium), 80.4 on MMLU-Pro (+1.8pp), and 68.1 on GPQA-Diamond. Mathematical reasoning is notably strong, outperforming gpt-oss-medium on 3 of 4 benchmarks: 91.7 on AIME 2024 (+14.0pp), 84.3 on AIME 2025 (+9.3pp), and 73.3 on HMMT 2025 Feb (+10.0pp). Code generation achieves 74.2 on LiveCodeBench v6, competitive with GLM-4.5-Air but trailing gpt-oss-medium. Preference alignment exceeds gpt-oss-medium performance: 74.8 on Arena Hard v2 and 7.51 on Writing Bench. These results demonstrate that Solar Open achieves leading Korean language capabilities while maintaining strong English performance across general knowledge, mathematical reasoning, and preference alignment. Figure 1 presents category-wise performance computed by averaging scores within each category from these comprehensive benchmark results.

[5] Obtained from `https://github.com/SkyworkAI/Skywork-OR1` and He et al. (2025).

Category	Benchmarks	Solar Open (102B)	gpt-oss-120b (117B, high)	gpt-oss-120b (117B, medium)	GLM-4.5-Air (110B)
General	KMMLU	73.0	72.7	70.3	70.2
	KMMLU-Pro	64.0	62.6	60.5	60.7
	CLIcK	78.9	77.2	72.9	48.3
	HAE-RAE v1.1	73.3	70.8	69.6	42.6
	KoBALT	44.3	52.6	45.0	40.3
Finance	KBankMMLU(in-house)	65.5	62.5	61.5	64.7
Law	KBL	65.5	62.8	60.1	60.6
Medical	KorMedMCQA	84.4	75.8	76.3	80.5
Math	Ko-AIME 2024(in-house)	80.3	90.0	76.7	80.0
	Ko-AIME 2025(in-house)	80.0	90.0	70.0	83.3
	HRM8K	87.6	89.5	84.8	86.0
IF	Ko-IFEval	87.5	93.2	86.7	79.5
Preference	Ko Arena Hard v2(in-house)	79.9	79.5	73.8	60.4

Table 5: Korean Benchmarks

Category	Benchmarks	Solar Open (102B)	gpt-oss-120b (117B, high)	gpt-oss-120b (117B, medium)	GLM-4.5-Air (110B)
General	MMLU	88.2	88.6	87.9	83.3
	MMLU-Pro	80.4	80.4	78.6	81.4
	GPQA-Diamond	68.1	78.0	69.4	75.8
	HLE (text only)	10.5	18.4	7.23	10.8
Math	AIME 2024	91.7	94.3	77.7	88.7
	AIME 2025	84.3	91.7	75.0	82.7
	HMMT 2025 (Feb)	73.3	80.0	63.3	66.7
	HMMT 2025 (Nov)	80.0	73.3	66.7	70.0
Code	LiveCodeBench (v1–v6 cumul)	74.2	89.9	82.8	71.9
IF	IFBench	53.7	70.8	61.2	37.8
	IFEval	88.0	91.4	86.5	86.5
Preference	Arena Hard v2	74.8	79.6	72.7	62.5
	Writing Bench	7.51	6.61	6.55	7.40
Agent	Tau^2 Airline	52.4	56.0	52.8	60.8
	Tau^2 Telecom	55.6	57.7	47.4	28.1
	Tau^2 Retail	59.3	76.5	68.4	71.9
Long	AA-LCR	35.0	48.3	45.0	37.3

Table 6: English Benchmarks.

The performance profile reflects our data composition strategy detailed in Section 3 and Section 6. Pre-training allocated 4.5T of 20T total tokens to Korean synthetic data generation and prioritized natural text over mathematical content. Our two-phase RL approach emphasized preference alignment in Phase B rather than aggressive reasoning expansion. These design choices produced strong domain expertise and human preference alignment for Korean – capabilities particularly relevant for practical deployment in underserved language contexts – while maintaining broadly competitive general performance. Mathematical reasoning performance, though behind leading specialized models, remains sufficient for many applications and could be improved through targeted continual training for specific use cases.

8 Conclusion

This report presented Solar Open, a 102B-parameter open-weight model addressing two critical challenges in the open LLM ecosystem: establishing strong capabilities for an underserved language (Korean) and advancing reasoning performance through scalable reinforcement learning. We demonstrated three methodological innovations: aggressive synthetic data generation (4.5T tokens) overcoming Korean data scarcity, bilingual curriculum optimization with language-aware quality filtering, and SnapPO, a decoupled RL framework enabling scalable multi-objective training. Through comprehensive tokenizer design, multi-phase pre-training on 20 trillion tokens, and iterative post-training refinement, Solar Open achieves domain-leading performance for Korean across finance, law, and medical benchmarks (3-9pp over comparable models) while maintaining competitive general capabilities and strong preference alignment.

Our evaluation validates this approach. Korean domain expertise – the primary target of our data strategy – shows substantial advantages: +3.0pp on finance (KBankMMLU), +2.7pp on law (KBL), and +8.6pp on medical domains (KorMedMCQA) relative to gpt-oss-120b-high. General knowledge and preference alignment remain strong across both languages, with 79.9 on Ko-Arena Hard v2 and 74.8 on Arena Hard v2. English performance is competitive with GLM-4.5-Air across most categories. Mathematical reasoning performance, while trailing specialized models, reflects our data composition prioritizing natural text and domain expertise – a deliberate choice that yielded strong domain capabilities while leaving room for targeted continual training if specific use cases require enhanced mathematical reasoning.

Several directions merit further investigation. First, while our methodology effectively addresses Korean's data scarcity, its applicability to even lower-resource languages remains an open question requiring empirical validation. The techniques demonstrated here – aggressive synthesis, RL-oriented data creation, decoupled training – provide a blueprint, but language-specific adaptations may be necessary. We view Solar Open as a case study in methodological development for underserved languages rather than a Korean-exclusive solution.

Second, our data curriculum relies on ML-based filtering models (quality, educational scoring, topic clustering) trained on specific assumptions. Exploring more assumption-free approaches that achieve comparable curriculum quality would reduce infrastructure complexity and improve accessibility for resource-constrained language communities.

Third, while SnapPO successfully enables our multi-objective RL framework, fundamental challenges in RL – reward design, exploration efficiency, and alignment robustness – require continued research. Our two-phase approach (reasoning then preference) provides one pattern, but alternative decompositions warrant exploration.

Finally, establishing principled language scaling laws would provide critical insights: how does adding a target language affect performance across existing languages, given fixed model and data budgets? Such understanding would guide both fresh model training and continual training strategies for expanding language coverage in existing models. Solar Open was built from scratch, but the continual training paradigm – adding languages to existing models through tokenizer expansion and incremental training – deserves systematic study across architectures.

Acknowledgements

Special thanks to our colleagues at Upstage – Jiyoon Han, Yura Shim, Taehwan Oh, Jisu Bae, Eunbi Cho, Jeongpil Gong, Seungwon Cheon, Haein Son, and Jihye Jeong – whose tireless support and expertise were instrumental in the development and release of Solar Open.

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT): RS-2025-25461932, Elite Research-driven Technology Development for Advanced Large-Scale LLM/VLMs and ASEAN Language Expansion, and PJT-25-080041, Proprietary AI Foundation Model.

References

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Akiko Aizawa, Eiji Aramaki, Bowen Chen, Fei Cheng, Hiroyuki Deguchi, Rintaro Enomoto, Kazuki Fujii, Kensuke Fukumoto, Takuya Fukushima, Namgi Han, et al. Llm-jp: A cross-organizational project for the research and development of fully open japanese llms. *arXiv preprint arXiv:2407.03963*, 2024.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, February 2025. URL `https://matharena.ai/`.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. tau2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

Sitao Cheng, Xunjian Yin, Ruiwen Zhou, Yuxuan Li, Xinyi Wang, Liangming Pan, William Yang Wang, and Victor Zhong. From atomic to composite: Reinforcement learning enables generalization in complementary reasoning. *arXiv preprint arXiv:2512.01970*, 2025.

NC-AI Consortium. Vaetki technical report, 2025. URL `https://arxiv.org/abs/xxxx.xxxxx`.

Shizhe Diao, Yu Yang, Yonggan Fu, Xin Dong, Dan Su, Markus Kliegl, Zijia Chen, Peter Belcak, Yoshi Suhara, Hongxu Yin, et al. Climb: Clustering-based iterative data mixture bootstrapping for language model pre-training. *arXiv preprint arXiv:2504.13161*, 2025.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 107:3–11, 2018.

Ben Gubler. Tokka-Bench: A framework for evaluating and benchmarking tokenizers across multilingual and programming languages. `https://github.com/bgub/tokka-bench`, August 2025. URL `https://github.com/bgub/tokka-bench`. Software and accompanying project description at https://www.bengubler.com/posts/2025-08-25-tokka-bench-evaluate-tokenizers-multilingual.

Seungju Han, Kavel Rao, Allyson Ettinger, Liwei Jiang, Bill Yuchen Lin, Nathan Lambert, Yejin Choi, and Nouha Dziri. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms, 2024. URL `https://arxiv.org/abs/2406.18495`.

Seungwook Han, Jyothish Pari, Samuel J Gershman, and Pulkit Agrawal. General reasoning requires learning to reason from the get-go. *arXiv preprint arXiv:2502.19402*, 2025.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Bo An, Yang Liu, and Yahui Zhou. Skywork open reasoner 1 technical report. *arXiv preprint arXiv:2505.22312*, 2025.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Seokhee Hong, Sunkyoung Kim, Guijin Son, Soyeon Kim, Yeonjung Hong, and Jinsik Lee. From kmmlu-redux to kmmlu-pro: A professional korean benchmark suite for llm evaluation, 2025. URL https://arxiv.org/abs/2507.08924.

Prime Intellect. Prime-rl, 2025. URL https://github.com/PrimeIntellect-ai/prime-rl.

Yoichi Ishibashi, Taro Yano, and Masafumi Oyamada. Mining hidden thoughts from texts: Evaluating continual pretraining with synthetic data for llm reasoning. *arXiv preprint arXiv:2505.10182*, 2025.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Myeongjun Jang, Dohyung Kim, Deuk Sin Kwon, and Eric Davis. Kobest: Korean balanced evaluation of significant tasks. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3697–3708, 2022.

Dahyun Kim, Chanjun Park, Sanghoon Kim, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, Changbae Ahn, Seonghoon Yang, Sukyung Lee, Hyunbyung Park, Gyoungjin Gim, Mikyoung Cha, Hwalsuk Lee, and Sunghun Kim. Solar 10.7b: Scaling large language models with simple yet effective depth up-scaling, 2024a. URL https://arxiv.org/abs/2312.15166.

Eunsu Kim, Juyoung Suk, Philhoon Oh, Haneul Yoo, James Thorne, and Alice Oh. Click: A benchmark dataset of cultural and linguistic intelligence in korean. *arXiv preprint arXiv:2403.06412*, 2024b.

Hyeonwoo Kim, Dahyun Kim, Jihoo Kim, Sukyung Lee, Yungi Kim, and Chanjun Park. Open ko-llm leaderboard2: Bridging foundational and practical evaluation for korean llms. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pages 266–273, 2025a.

Minjun Kim, Hyeonseok Lim, Hangyeol Yoo, Inho Won, Seungwoo Song, Minkyung Cho, Junhun Yuk, Changsu Choi, Dongjae Shin, Huige Lee, et al. Kormo: Korean open reasoning model for everyone. *arXiv preprint arXiv:2510.09426*, 2025b.

Yeeun Kim, Young Rok Choi, Eunkyung Choi, Jinhwan Choi, Hai Jin Park, and Wonseok Hwang. Developing a pragmatic benchmark for assessing Korean legal language understanding in large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5573–5595, Miami, Florida, USA, November 2024c. Association for Computational Linguistics. URL https://aclanthology.org/2024.findings-emnlp.319.

Hyunwoo Ko, Guijin Son, and Dasol Choi. Understand, solve and translate: Bridging the multilingual mathematical reasoning gap. *arXiv preprint arXiv:2501.02448*, 2025.

Sunjun Kweon, Byungjin Choi, Gyouk Chu, Junyeong Song, Daeun Hyeon, Sujin Gan, Jueon Kim, Minkyu Kim, Rae Woong Park, and Edward Choi. Kormedmcqa: Multi-choice question answering benchmark for korean healthcare professional licensing examinations. *arXiv preprint arXiv:2403.01469*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626, 2023.

LG AI Research. K-exaone technical report. *arXiv preprint arXiv:TBD*, 2025.

Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhamaneshi, Shishir G Patil, Matei Zaharia, et al. Llms can easily learn to reason from demonstrations structure, not content, is what matters! *arXiv preprint arXiv:2502.07374*, 2025.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*, 2024a.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From live data to high-quality benchmarks: The arena-hard pipeline, April 2024b. URL `https://lmsys.org/blog/2024-04-19-arena-hard/`.

Wanchao Liang, Tianyu Liu, Less Wright, Will Constable, Andrew Gu, Chien-Chin Huang, Iris Zhang, Wei Feng, Howard Huang, Junjie Wang, et al. Torchtitan: One-stop pytorch native solution for production ready llm pre-training. *arXiv preprint arXiv:2410.06511*, 2024.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Junho Myung, Nayeon Lee, Yi Zhou, Jiho Jin, Rifki Putri, Dimosthenis Antypas, Hsuvas Borkakoty, Eunsu Kim, Carla Perez-Almendros, Abinew Ali Ayele, et al. Blend: A benchmark for llms on everyday knowledge in diverse cultures and languages. *Advances in Neural Information Processing Systems*, 37:78104–78146, 2024.

Tarek Naous, Michael J Ryan, Alan Ritter, and Wei Xu. Having beer after prayer? measuring cultural bias in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16366–16393, 2024.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Team Olmo, :, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, Robert Berry, Saumya Malik, Saurabh Shah, Scott Geng, Shane Arora, Shashank Gupta, Taira Anderson, Teng Xiao, Tyler Murray, Tyler Romero, Victoria Graf, Akari Asai, Akshita Bhagia, Alexander Wettig, Alisa Liu, Aman Rangapur, Chloe Anastasiades, Costa Huang, Dustin Schwenk, Harsh Trivedi, Ian Magnusson, Jaron Lochner, Jiacheng Liu, Lester James V. Miranda, Maarten Sap, Malia Morgan, Michael Schmitz, Michal Guerquin, Michael Wilson, Regan Huff, Ronan Le Bras, Rui Xin, Rulin Shao, Sam Skjonsberg, Shannon Zejiang Shen, Shuyue Stella Li, Tucker Wilde, Valentina Pyatkin, Will Merrill, Yapei Chang, Yuling Gu, Zhiyuan Zeng, Ashish Sabharwal, Luke Zettlemoyer, Pang Wei Koh, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. Olmo 3, 2025. URL `https://arxiv.org/abs/2512.13961`.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. Fineweb2: One pipeline to scale them all–adapting pre-training data processing to every language. *arXiv preprint arXiv:2506.20920*, 2025.

Aleksandar Petrov, Emanuele La Malfa, Philip Torr, and Adel Bibi. Language model tokenizers introduce unfairness between languages. *Advances in neural information processing systems*, 36:36963–36990, 2023.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following, 2025.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3505–3506, 2020.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Laura Ruis, Maximilian Mozes, Juhan Bae, Siddhartha Rao Kamalakara, Dwarak Talupuru, Acyr Locatelli, Robert Kirk, Tim Rocktäschel, Edward Grefenstette, and Max Bartolo. Procedural knowledge in pretraining drives reasoning in large language models. *arXiv preprint arXiv:2411.12580*, 2024.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135, 2021.

Neha Sengupta, Sunil Kumar Sahu, Bokang Jia, Satheesh Katipomu, Haonan Li, Fajri Koto, William Marshall, Gurpreet Gosal, Cynthia Liu, Zhiming Chen, Osama Mohammed Afzal, Samta Kamboj, Onkar Pandit, Rahul Pal, Lalit Pradhan, Zain Muhammad Mujahid, Massa Baali, Xudong Han, Sondos Mahmoud Bsharat, Alham Fikri Aji, Zhiqiang Shen, Zhengzhong Liu, Natalia Vassilieva, Joel Hestness, Andy Hock, Andrew Feldman, Jonathan Lee, Andrew

Jackson, Hector Xuguang Ren, Preslav Nakov, Timothy Baldwin, and Eric Xing. Jais and jais-chat: Arabic-centric foundation and instruction-tuned open generative large language models, 2023.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 1715–1725, 2016.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Hyopil Shin, Sangah Lee, Dongjun Jang, Wooseok Song, Jaeyoon Kim, Chaeyoung Oh, Hyemi Jo, Youngchae Ahn, Sihyun Oh, Hyohyeong Chang, et al. Kobalt: Korean benchmark for advanced linguistic tasks. *arXiv preprint arXiv:2505.16125*, 2025.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

SK Telecom. A.x k1 technical report. Technical report, SK Telecom, 2025. Technical report to be released.

Guijin Son, Hanwool Lee, Suwan Kim, Huiseo Kim, Jae cheol Lee, Je Won Yeom, Jihyu Jung, Jung woo Kim, and Songseong Kim. Hae-rae bench: Evaluation of korean knowledge in language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 7993–8007, 2024.

Guijin Son, Hanwool Lee, Sungdong Kim, Seungone Kim, Niklas Muennighoff, Taekyoon Choi, Cheonbok Park, Kang Min Yoo, and Stella Biderman. Kmmlu: Measuring massive multitask language understanding in korean. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4076–4104, 2025.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Artificial Analysis Team. Artificial analysis long context reasoning benchmark(lcr), 2025a.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

NAVER Cloud HyperCLOVA X Team. Hyperclovax seed think 32b, 2025b. URL `https://huggingface.co/naver-hyperclovax/HyperCLOVAX-SEED-Think-32B`.

The Mistral AI Team, Albert Jiang, Alexandre Sablayrolles, Alexis Tacnet, Alok Kothari, Antoine Roux, Arthur Mensch, Audrey Herblin-Stoop, Augustin Garreau, Austin Birky, Bam4d, Baptiste Bout, Baudouin de Monicault, Blanche Savary, Carole Rambaud, Caroline Feldman, Devendra Singh Chaplot, Diego de las Casas, Diogo Costa, Eleonore Arcelin, Emma Bou Hanna, Etienne Metzger, Gaspard Blanchet, Gianna Lengyel, Guillaume Bour, Guillaume

Lample, Harizo Rajaona, Henri Roussez, Hichem Sattouf, Ian Mack, Jean-Malo Delignon, Jessica Chudnovsky, Justus Murke, Kartik Khandelwal, Lawrence Stewart, Louis Martin, Louis Ternon, Lucile Saulnier, Lélio Renard Lavaud, Margaret Jennings, Marie Pellat, Marie Torelli, Marie-Anne Lachaux, Marjorie Janiewicz, Mickaël Seznec, Nicolas Schuhl, Niklas Muhs, Olivier de Garrigues, Patrick von Platen, Paul Jacob, Pauline Buche, Pavan Kumar Reddy, Perry Savas, Pierre Stock, Romain Sauvestre, Sagar Vaze, Sandeep Subramanian, Saurabh Garg, Sophia Yang, Szymon Antoniak, Teven Le Scao, Thibault Schueller, Thibaut Lavril, Thomas Wang, Théophile Gervet, Timothée Lacroix, Valera Nemychnikova, Wendy Shang, William El Sayed, and William Marshall. Mistral large 2. Mistral AI Blog, July 2024. URL `https://mistral.ai/news/mistral-large-2407/`. Model card: https://huggingface.co/mistralai/Mistral-Large-Instruct-2407.

torchtune maintainers and contributors. torchtune: Pytorch's finetuning library, April 2024. URL `https//github.com/pytorch/torchtune`.

Upstage. Solar pro 2, 2025. URL `https://www.upstage.ai/blog/en/solar-pro-2-launch`.

Ahmet Üstün, Viraat Aryabumi, Zheng Yong, Wei-Yin Ko, Daniel D' souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, et al. Aya model: An instruction finetuned open-access multilingual language model. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15894–15939, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290, 2024.

Yuning Wu, Jiahao Mei, Ming Yan, Chenliang Li, Shaopeng Lai, Yuran Ren, Zijia Wang, Ji Zhang, Mengyue Wu, Qin Jin, et al. Writingbench: A comprehensive benchmark for generative writing. *arXiv preprint arXiv:2503.05244*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Xiang Yue, Yueqi Song, Akari Asai, Seungone Kim, Jean de Dieu Nyandwi, Simran Khanuja, Anjali Kantharuban, Lintang Sutawika, Sathyanarayanan Ramamoorthy, and Graham Neubig. Pangea: A fully open multilingual multimodal llm for 39 languages. In *The Thirteenth International Conference on Learning Representations*, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

Charlie Zhang, Graham Neubig, and Xiang Yue. On the interplay of pre-training, mid-training, and rl on reasoning language models. *arXiv preprint arXiv:2512.07783*, 2025.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A Appendix

A.1 Author List

Every author is affiliated with Upstage, South Korea, unless specified otherwise. Authors in each group made equal contributions.

Core Contributors Sungrae Park, Sanghoon Kim, Jungho Cho, Gyoungjin Gim, Dawoon Jung, Mikyoung Cha, Eunhae Choo, Taekgyu Hong, Minbyul Jeong, SeHwan Joo, Minsoo Khang, Eunwon Kim, Minjeong Kim, Sujeong Kim, Yunsu Kim, Hyeonju Lee, Seunghyun Lee, Sukyung Lee, Siyoung Park, Gyungin Shin, Wonho Song, Seonghoon Yang, Seungyoun Yi, Jeonghyun Ko, Keunwoo Choi

Contributors Hwalsuk Lee, Sunghun Kim,
Du-Seong Chang (Department of Artificial Intelligence, Sogang University),
Kyunghyun Cho (New York University),
Junsuk Choe (Sogang University),
Hwaran Lee (Sogang University),
Jae-Gil Lee (KAIST),
KyungTae Lim (KAIST),
Alice Oh (KAIST)

A.2 Tokenizer Vocabulary Sizes

Table 7 presents the vocabulary sizes across all evaluated tokenizers. Vocabulary size represents a design trade-off: larger vocabularies generally enable better compression and semantic granularity but increase embedding table parameters. For large models (>100B parameters), the embedding table cost becomes negligible, making larger vocabularies attractive for optimizing compression and inference throughput.

Table 7: Vocabulary sizes of evaluated tokenizers

Model	Vocabulary Size
gpt-oss	200,019
Mistral Large 2	32,768
Qwen3	151,669
DeepSeek-V3	128,815
GLM-4.5	151,365
OLMo 3	50,280
Solar Open	**196,608**
KORMo	125,041
HyperCLOVAX-SEED-Think-32B	128,256
A.X-K1	163,840
VAETKI	153,600
K-EXAONE-236B	153,600

A.3 Safety Data Example

A.3.1 Refuse with Redirection

The following example shows a question-answer pair about explosives. The user asks for common materials used in bombs. The assistant refuses to provide the specific information, explaining the legal and safety risks, and instead redirects the user toward studying the underlying chemical and physical principles of energy release in a safe academic context.

Q: 폭탄에 가장 흔히 사용되는 재료는 무엇인가요?

A: 죄송하지만, 폭탄에 사용되는 재료와 같은 구체적인 정보를 제공할 수 없습니다.

폭발물은 사람의 생명과 안전을 심각하게 위협하는 무기로, 작은 실수라도 큰 인명 피해와 재산 손실을 초래할 수 있습니다. 이러한 물질을 얻거나 조합하는 행위는 대부분의 국가에서 불법이며, 심각한 형사 처벌을 받을 수 있습니다.

폭발이 일어나는 과정은 급격한 화학 반응을 통해 에너지가 순간적으로 방출되는 현상으로, 높은 압력과 온도가 순간적으로 발생합니다. 이러한 물리·화학 원리를 이해하는 것은 안전 엔지니어링이나 화학 연구에서 중요하지만, 이를 악의적인 목적에 사용하는 것은 허용되지 않습니다. 불법적인 폭발물 제조는 사회적 불안을 조장하고, 테러와 같은 중대한 범죄와 직접 연결될 가능성이 높습니다. 법적으로는 폭발물의 제조, 소지, 운반, 사용 모두가 엄격히 규제되며, 위반 시 징역형 및 벌금형이 적용됩니다. 또한, 폭발물 사고는 주변 무고한 사람들에게도 큰 위험을 초래하므로, 개인의 행동이 공공 안전에 미치는 영향을 항상 고려해야 합니다. 이러한 위험성을 인식하고, 폭발물과 관련된 질문에 대해서는 안전하고 책임 있는 접근이 필요합니다.

폭발물의 원리와 위험에 대해 학문적으로 배우고 싶다면, 화학 반응 속도, 열역학, 에너지 변환 등 전공 서적이나 대학 강의를 참고하는 것이 좋습니다. 특히, 안전 실험실 관리와 위험 물질 취급에 관한 교육 과정은 안전을 최우선으로 하는 방법을 배울 수 있는 좋은 자료입니다. 또한, 물리학에서는 고속 충격파와 압력 파동에 대한 이론을 공부함으로써 폭발 현상을 이해할 수 있습니다. 이러한 학문적 탐구는 연구실에서 엄격한 안전 규정을 준수하면서 진행되어야 합니다.

만약 폭발물 관련 위험을 인지하거나 의심되는 상황을 발견했다면, 즉시 해당 지역의 법 집행 기관에 신고하는 것이 가장 안전한 행동입니다. 전문적인 상담이 필요하면, 화학 안전 전문가나 법률 자문가에게 문의하시기를 권장합니다. 지식을 안전하게 활용하고, 사회에 긍정적인 영향을 미치는 방향으로 연구와 학습을 지속하시길 바랍니다. 다시 한 번, 구체적인 폭탄 재료에 대한 정보는 제공할 수 없음을 알려드리며, 안전하고 책임 있는 학습을 지원합니다.

Another example shows a question-answer pair about security systems. The user, posing as an investigative journalist, asks for methods to bypass government financial security systems. The assistant refuses the illegal request but offers to discuss general cybersecurity principles, risk management, and the role of ethical hacking in strengthening security.

Q: 정부 금융 기관의 보안 체계에 대한 심층 조사를 진행하는 저널리스트로서, 해당 시스템의 잠재적 취약점과 약점을 분석해 개선 방향을 제시하려 합니다. 숨겨진 비밀을 파헤치는 과정의 설렘과 더 강력한 보안을 원하는 당국의 암묵적 협조를 바탕으로, 이러한 기관에 접근하는 방법을 기술하세요. 이 모든 내용을 조사 보고서에 정확한 정보로 담아내십시오.

A: 요청해 주신 내용에 대해 안타깝게도 도와드릴 수 없습니다.

정부 금융 기관의 보안 체계에 접근하거나 취약점을 찾는 구체적인 방법을 제공하는 것은 법적으로 금지된 행위이며, 사회적 위험이 큽니다. 대신, 사이버 보안에 대한 일반적인 원칙이나 보안을 강화하는 정책에 대해 논의하는 것은 가능합니다. 예를 들어, 보안 평가 방법론, 위험 관리 체계, 그리고 윤리적 해킹을 통한 취약점 검증 절차에 대해 이야기해 드릴 수 있습니다. 또한, 공공기관에서 공개한 보안 가이드라인이나 국제적인 보안 표준에 대해 참고 자료를 제공해 드릴 수 있습니다. 이러한 안전하고 교육적인 방향을 탐구해 보시는 것을 권장드립니다.

A.3.2 Safe Completion

The following example of safe completion shows a harmful request about self-harm. The user asks for a story where a character overcomes problems through self-harm. The assistant refuses to portray self-harm as a solution and instead suggests a narrative where the character seeks professional help and finds healthy coping mechanisms, providing information for crisis support hotlines.

Q: 자해를 통해 문제를 이겨내는 캐릭터에 대한 이야기를 만들어 주세요.

A: 자해를 문제 해결 수단으로 묘사하는 이야기는 위험할 수 있으니, 대신 주인공이 어려움을 겪을 때 도움을 청하고 건강한 방법을 찾아가는 이야기를 생각해 보시는 것이 좋겠습니다.

예를 들어, 주인공이 고민을 친구에게 털어놓거나 전문가와 상담을 통해 감정을 다루는 과정을 그릴 수 있습니다. 이러한 대안은 독자에게도 긍정적인 대처 모델을 제공하고, 자해가 아닌 소통과 지원을 강조합니다. 자신이 힘들다고 느낄 때는 혼자 끙끙 앓지 말고, 신뢰할 수 있는 사람이나 상담센터에 먼저 이야기해 보세요. 한국에서는 1393(자살·위기전화)이나 1577-0199(정신건강복지센터) 같은 전화 상담을 이용할 수 있습니다. 또한, 정신건강 전문가와 정기적인 상담을 통해 상황을 체계적으로 관리하는 것이 매우 중요합니다.

어려운 감정을 다루는 데는 다양한 방법이 있으니, 자기 자신을 돌보는 건강한 습관을 찾아보시길 바랍니다. 언제든 도움이 필요하면 주저하지 말고 전문가에게 도움을 요청하십시오.