Kanana: Compute-efficient Bilingual Language Models

Kanana LLM Team *
kanana-llm@kakaocorp.com

https://huggingface.co/kakaocorp
https://github.com/kakao/kanana

Abstract

We introduce Kanana, a series of bilingual language models that demonstrate exceeding performance in Korean and competitive performance in English. The computational cost of Kanana is significantly lower than that of state-of-the-art models of similar size. The report details the techniques employed during pre-training to achieve compute-efficient yet competitive models, including high quality data filtering, staged pre-training, depth up-scaling, and pruning and distillation. Furthermore, the report outlines the methodologies utilized during the post-training of the Kanana models, encompassing supervised fine-tuning and preference optimization, aimed at enhancing their capability for seamless interaction with users. Lastly, the report elaborates on plausible approaches used for language model adaptation to specific scenarios, such as embedding, retrieval augmented generation, and function calling. The Kanana model series spans from 2.1B to 32.5B parameters with 2.1B models (base, instruct, embedding) publicly released to promote research on Korean language models.

1 Introduction

Recent breakthroughs in large language models (LLMs) have been driven by increasing training data (Hoffmann et al., 2022) and model parameters (Brown et al., 2020; Kaplan et al., 2020; Chowdhery et al., 2023). However, advances have also introduced substantial computational costs that reach millions of dollars (Grattafiori et al., 2024), which poses a challenge to the community on developing LLMs *from scratch*. As a result, reducing computational cost has emerged as a crucial problem in order to popularize the development of LLMs for both academia and industry (Zhao et al., 2024; Fishman et al., 2025; Wang et al., 2025). To this end, recent works have presented various solutions to the computation problem in model architectures and scaling (Shao et al., 2024a; Kim et al., 2024a; Muralidharan et al., 2024), through data (Penedo et al., 2024a; Sachdeva et al., 2024), and through training strategies (DeepSeek-AI, 2024; Hu et al., 2024).

As the product of our endeavor to address the computational challenges, we introduce *Kanana* model family, developed using only a fraction of computational cost while maintaining performance compared to those of the state-of-the-art (SOTA) open LLMs. The family of models includes pre-trained base model and post-trained instruction models in sizes of {2.1B, 9.8B, 32.5B}. We show in Figure 1 that Kanana models establish a new Pareto frontier in the computational cost of the train time versus the performance.

In the pre-training phase, as it accounts for the majority of the training costs for LLMs, we focus on reducing its computational demands while maintaining performance. Since the cost of the pre-training phase primarily arises from the large dataset size and model scale, we reduce it by improving both data efficiency and training efficiency. To improve data efficiency, we carefully curate a training dataset of 3 trillion tokens, enabling our models to achieve competitive performance despite using a smaller dataset than SOTA pre-trained models. For training efficiency, we employ cost-effective techniques such as staged pre-training (Hu et al., 2024; Ibrahim et al., 2024) and depth up-scaling (Kim et al., 2024a) to reduce computational costs associated with model size. From the models obtained, we

*A detailed contributor list can be found in the last section of the main paper.

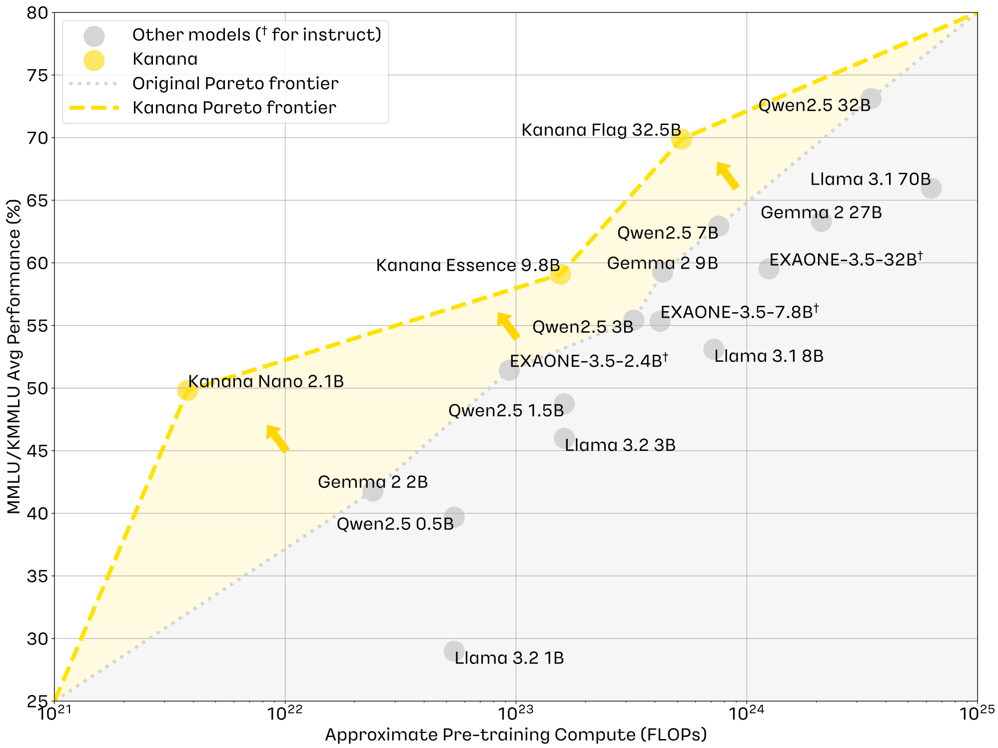


Figure 1: Performance to pre-training computational cost for Kanana and comparable models. We measure computational cost in FLOPs (Floating Point Operations), which is approximately calculated as 6 $\times$ training tokens $\times$ model size (Kaplan et al., 2020). We only calculate student training FLOPs for distillation models. Obviously, Kanana models achieves decent performance given their limited computational cost.

extend pruning and distillation technique (Muralidharan et al., 2024) to train smaller models using only a handful subset of the pre-training data.

Leveraging the strong performances of Kanana base models, we further develop instruction and domain-specific adaptation models. To develop instruction models, we apply a post-training process that includes supervised fine-tuning and preference optimization. As a result, our instruction models achieve competitive performance to that of SOTA models on various tasks, including English/Korean chat, general knowledge reasoning, instruction following, code generation, and mathematical problem-solving. In addition, we adapt instruction models to develop embedding models, retrieval-augmented generation models, and function-calling models.

2 Pre-training

Since pre-training constitutes the majority of computational costs, we focus on reducing the expenses of this stage and show our results in Section 2.1. To enhance efficiency in pre-training LLMs, we employ two key strategies: data efficiency and training efficiency. In Section 2.2, we discuss our data curation method to maximize the data efficiency under fixed token budget. In Section 2.3, we adopt cost-effective training techniques to minimize the computational overhead associated with model scaling.

2.1 Performance

We evaluate our pre-trained models using a series of standard benchmarks designed to assess English/Korean general knowledge, code, and mathematical reasoning. For general

Models	MMLU *5-shot*	KMMLU *5-shot*	HAE-RAE *5-shot*	HumanEval *0-shot*	MBPP *3-shot*	GSM8K *5-shot*	Avg
Kanana Flag 32.5B	77.68	62.10	**90.47**	**51.22**	63.40	70.05	69.15
Qwen2.5 32B	**83.10**	**63.15**	75.16	50.00	**73.40**	**82.41**	71.20
Gemma 2 27B	75.45	51.16	69.11	**51.22**	64.60	74.37	64.32
EXAONE-3.5-32B†	72.68	46.36	82.22	-	-	-	-
Aya Expanse 32B†	74.52	49.57	80.66	-	-	-	-
Kanana Essence 9.8B	67.61	50.57	**84.97**	40.24	53.60	63.61	60.10
Llama 3.1 8B	65.18	41.02	61.78	35.37	48.60	50.87	50.47
Qwen2.5 7B	**74.19**	**51.68**	67.46	**56.71**	**63.20**	**83.85**	66.18
Gemma 2 9B	70.34	48.18	66.18	37.20	53.60	68.16	57.28
EXAONE-3.5-7.8B†	65.36	45.30	77.54	-	-	-	-
Aya Expanse 8B†	62.52	40.11	71.95	-	-	-	-
Kanana Nano 2.1B	54.83	44.80	**77.09**	31.10	46.20	46.32	50.06
Llama 3.2 3B	56.40	35.57	47.66	25.61	39.00	27.37	38.60
Qwen2.5 3B	**65.57**	**45.28**	61.32	**37.80**	**55.60**	**69.07**	55.77
Gemma 2 2B	52.89	30.67	45.55	20.12	28.20	24.72	33.69
EXAONE-3.5-2.4B†	59.27	43.58	68.65	-	-	-	-
Llama 3.1 70B	78.93	53.00	76.35	57.32	66.60	81.73	68.99
Qwen2.5 72B	86.12	68.57	80.84	55.49	76.40	92.04	76.58

Table 1: Performance of Kanana base models on a set of standard benchmarks. The best scores are denoted in bold. 70B sized Models have been included for reference purposes. † For these models, results are obtained using instruct models because base model checkpoints are not released.

knowledge, we employ multiple choice tasks of MMLU (Hendrycks et al., 2021a) for English knowledge, and KMMLU (Son et al., 2024a) and HAE-RAE (Son et al., 2024b) for Korean-specific knowledge. To evaluate domain-specific abilities, we use HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) for code and GSM8K (Cobbe et al., 2021) for mathematical reasoning. We use log-likelihood for multiple choice tasks, and greedy generation for generative tasks.

To demonstrate the effectiveness of our training strategy, we compare our models with representative open-source models in various model sizes (Grattafiori et al., 2024; Team et al., 2024b; Qwen et al., 2025; Research, 2024b; Dang et al., 2024). For EXAONE and Aya Expanse models (Research, 2024b; Dang et al., 2024), we report only the performances on multiple-choice tasks using the same evaluation protocol. This decision is based on the observation that multiple-choice performances largely remain unchanged between the base and instruct models, whereas generative tasks exhibit notable divergences (see Appendix A.1 for a detailed discussion).

As shown in Table 1 and Figure 1, our models demonstrate strong performance in various domains and exhibit impressive Korean language capabilities, while requiring significantly less training compute. Kanana Flag 32.5B outperforms Llama 3.1 70B, Gemma 2 27B, and EXAONE-3.5-32B on knowledge-intensive natural language understanding benchmarks, such as MMLU and KMMLU, while consuming substantially fewer computational resources. In particular, the computational cost is even lower than that of Llama 3.1 8B, and is similar to Gemma 2 9B and EXAONE-3.5-7.8B. On the HAE-RAE benchmark, all Kanana LLMs demonstrate superior performance compared to other LLMs of similar sizes.

2.2 Data

We train Kanana models on 3 trillion tokens, primarily focusing on English and Korean bilingual capabilities. We collect our corpora from various sources and categorize them as English web, Korean web, academic, code, encyclopedic documents, and instruction data. All our data come from publicly available sources and do not include data from Kakao's products or services.

We begin by collecting various open-source datasets from multiple high-quality sources such as arXiv and Wikipedia. However, we observe that these datasets often suffer quality issues due to suboptimal extraction pipelines, resulting in omissions or incoherent paragraph ordering (see Appendix A.2 for details). Inherently, we improve source-specific extraction processes for these sources and re-extract documents with more valuable information and higher coherence. For code datasets, we utilize open-source datasets from Li et al. (2023b) and Lozhkov et al. (2024). We use only permissively licensed code and exclude any with non-permissive or missing licenses. Following INF-Team (2024)'s observation that adding instruction data at the end of pre-training enhances performance after SFT, we also incorporate instruction data with decontamination.

Utilizing the high potential of web as a source of valuable and diverse documents (Li et al., 2024; Su et al., 2024; Shao et al., 2024b), we apply series of filtering methods to extract high quality data. The first filtering process is cascaded filtering pipeline (AI et al., 2025; Grattafiori et al., 2024; Li et al., 2024; Team et al., 2024a; Penedo et al., 2024a) consisting of deduplication, heuristic filtering, and personally identifiable information (PII) anonymization. After the cascaded filtering, we further apply language-specific model-based filtering on high quality documents (Su et al., 2024; Shao et al., 2024b; Li et al., 2024; Penedo et al., 2024a) separately on English and Korean. For English web documents, we utilize a DCLM (Li et al., 2024) classifier. For Korean web documents, due to the lack of publicly available high quality classifiers, we iteratively train edu filter as high quality classifier using `FastText` (Joulin et al., 2017) based on the FineWeb-Edu pipeline (Penedo et al., 2024a). When applying the FineWeb-Edu pipeline, we observe that most of the documents are classified as uneducational, leading to a distribution imbalance. To address this issue, we iteratively retrain the classifier by augmenting educational documents from the previous iteration.

To assess the quality of our edu filter and Korean web corpus, we perform experiments by continual pre-training Llama 3 8B with 25B tokens. As shown in Table 2, the quality of our Korean web corpus is comparable to that of FineWeb 2 (Penedo et al., 2024b), which is the largest open-source Korean corpus. Furthermore, when using our edu filter to extract high quality data from Korean web corpus, we observe a significant performance improvement in the experimental results through training. Interestingly, we observe that using high quality English data, regardless of the quality of Korean data, can improve the scores on Korean benchmarks such as KMMLU and HAE-RAE, as well as the English benchmark MMLU. The results from this experiment make a foundation of our intuition for data mixture strategy in the staged pre-training in the following section.

English Corpus	Korean Corpus	MMLU *5-shot*	KMMLU *5-shot*	HAE-RAE *5-shot*
-	-	65.14	40.29	61.23
DCLM random	FineWeb2 Korean	64.16	41.02	70.39
DCLM random	Our Korean web	63.59	41.41	71.31
DCLM random	Our Korean web w/ edu filter	63.47	43.60	74.89
DCLM high	FineWeb2 Korean	65.36	41.78	71.22
DCLM high	Our Korean web	64.80	41.96	72.59
DCLM high	Our Korean web w/ edu filter	**65.40**	**44.19**	**75.99**

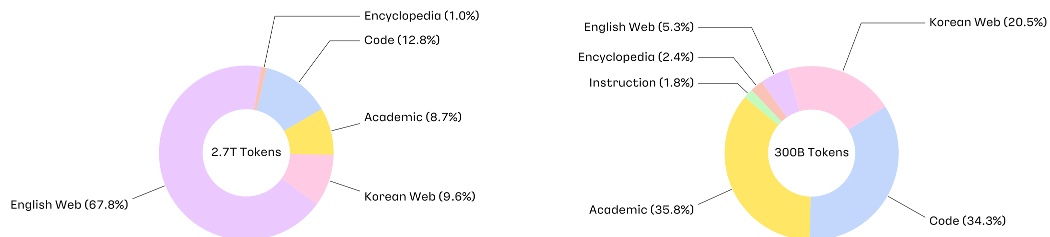
Table 2: Performance of Llama 3 8B before and after continual pre-training with only 25B tokens, using different combinations of English and Korean corpora at a 1:1 ratio.

In summary, we share two insights to consider when building bilingual corpora with underrepresented language for enhanced computational efficiency. (1) Prioritize quality over quantity. For languages that do not have vast tokens available, such as Korean, prioritizing quality over quantity is an effective solution. (2) Knowledge from English data transfers to Korean. Even with quality filtering on Korean dataset, English data remains a primary source of diverse and high-quality knowledge. We observe that, under the same conditions for the quality of Korean data, improving the quality of English data leads to higher scores on Korean-related benchmarks.

2.3 Training Process

To enhance computational efficiency in pre-training LLMs, we employ three key techniques: staged pre-training from scratch, depth up-scaling, and pruning and distillation. In Section 2.3.1, we first train 8B and 26.8B models using a staged pre-training approach, which serves as the foundation for obtaining LLMs at various scales. In Section 2.3.2, we describe the process to obtaining *Kanana Essence 32.5B* and *Kanana Flag 9.8B* models by depth up-scaling from 26.8B and 8B models, respectively. In Section 2.3.3, we derive *Kanana Nano 2.1B* model through pruning and distillation from the 8B model, reducing training costs while achieving superior performance compared to training a model from scratch.

2.3.1 Staged pre-training from scratch



(a) Stage 1 data (b) Stage 2 data

Figure 2: Kanana's staged pre-training data mixture.

Models	Stage	MMLU *5-shot*	KMMLU *5-shot*	HAE-RAE *5-shot*	HumanEval *0-shot*	MBPP *3-shot*	GSM8K *5-shot*	Avg
26.8B	Stage 1	73.38	54.26	84.97	32.32	47.20	57.77	58.32
	Stage 2	74.27	59.04	88.45	51.22	61.60	67.48	67.01
8B	Stage 1	63.48	45.51	77.27	23.78	35.80	35.03	46.81
	Stage 2	64.22	48.30	83.41	40.24	51.40	57.09	57.44

Table 3: Performance of from-scratch Kanana models at the end of each training stage.

To maximize performance under fixed compute budget, we adopt the staged pre-training strategy (Hu et al., 2024; Team et al., 2024a; Huang et al., 2024; Wake et al., 2025; Granite Team, 2024) with two stages. Staged pre-training divides the pre-training process into multiple stages, starting with training LLMs on a large amount of moderate-quality data in the initial stages, and gradually increasing the proportion of high quality data in the subsequent stages.

We begin by training 8B from scratch using the diverse 2.7 trillion in stage 1 as shown in Figure 2a. In stage 2, we further train the model using 300 billion tokens shown in Figure 2b. Specifically, we set aside high quality data for each category using the available high quality classifiers. Then, we perform lightweight annealing experiments to select candidate datasets to search for the data mixture following Grattafiori et al. (2024). Then, the optimal data mixture is selected through ablation study. The final model of stage 2 results in a 2.79 point increase in KMMLU and a 10.63 point increase in average performance, demonstrating the effectiveness and efficiency of staged pre-training. We apply the same data mixture that was used during the training of 8B to 26.8B model. Direct application of the recipe consistently yields remarkable performance and stable training as shown in Table 3, demonstrating the scalability of our recipe. See Appendix A.3 for our pre-training configurations.

2.3.2 Depth Up-scaling

To further enhance the model performance within limited resources after pre-training, we adopt the depth up-scaling (DUS) which increases model capacity by stacking additional layers (Kim et al., 2024a). We apply DUS to expand Kanana 8B into Kanana Essence 9.8B and Kanana 26.8B into Kanana Flag 32.5B. After the up-scaling process, each model variant is further trained on the same data mixtures used in pre-training, with 100 billion tokens dedicated to stage 1 and another 100 billion to stage 2. Results of the up-scaling strategy demonstrates that the additional layers consistently contribute to performance enhancements as summarized in Table 4.

Models	MMLU *5-shot*	KMMLU *5-shot*	HAE-RAE *5-shot*	HumanEval *0-shot*	MBPP *3-shot*	GSM8K *5-shot*	Avg
26.8B + DUS (32.5B)	77.68	62.10	90.47	51.22	63.40	70.05	69.15
26.8B	74.27	59.04	88.45	51.22	61.60	67.48	67.01
8B + DUS (9.8B)	67.61	50.67	84.98	40.24	53.60	63.61	60.10
8B	64.22	48.30	83.41	40.24	51.40	57.09	57.44

Table 4: Performance comparison of Kanana models before and after depth up-scaling.

Table 4 illustrates the performance improvements achieved through depth up-scaling. Kanana Essence 9.8B consistently outperforms its non-upscaled version, Kanana 8B with the average score rising from 57.52 to 60.12. This improvement is evident in MMLU, KMMLU, HAE-RAE, MBPP, and GSM8K, except for HumanEval. Similarly, Kanana Flag 32.5B achieves average score of 69.15, notably surpassing the non-upscaled Kanana 26.8B model. These results emphasize the effectiveness of depth up-scaling in improving various benchmark scores.

Notably, our strategy saves 11.06% of total computational cost compared to the training of 9.8B and 32.5B LLMs from scratch. This strategy of increasing model capacity through depth up-scaling only occupies about 6.67% of the total computing resources across the entire training procedure. In combination with pre-training, depth up-scaling offers a strategic approach to significantly enhance model performance without introducing heavy computational demands of building new models from scratch.

2.3.3 *Pruning and Distillation*

In opposition to efficiently up-scaling the model size, knowledge distillation is an effective method to efficiently down-scale the model size (Hinton et al., 2015; Gunter et al., 2024; Meta, 2024). Leveraging the 8B model from Section 2.3.1, we efficiently produce smaller models by improving the pruning and distillation of Minitron (Muralidharan et al., 2024; Sreenivas et al., 2024). This process allows us to produce models with better performance at one-tenth of the data size compared to training from scratch, as shown in Table 5. We further show that iteratively extending the process beyond two iterations remains effective, preserving 87-99% of KMMLU score at only 50% of the model size, as shown in Table 6. Our models achieve competitive performance to recent open-source models (Allal et al., 2025; Grattafiori et al., 2024; Team et al., 2024b; Qwen et al., 2025), as presented in Table 14.

Models	Training Tokens	MMLU *5-shot*	KMMLU *5-shot*	HAERAE *5-shot*	HumanEval *0-shot*	MBPP *3-shot*	GSM8K *5-shot*	Avg
2.1B PD	0.3T	54.83	44.80	77.09	31.10	46.20	46.32	50.06
2.1B	3T	50.66	36.61	68.74	24.45	41.60	36.69	43.13

Table 5: Token consumption and performance of pruning & distillation (PD) from preceding models and training from scratch. We use the same 2.1B architecture.

In order to improve the pruning and distillation process, we refine Minitron's width importance scoring while preserving its simplicity and efficiency. Its scoring process begins by measuring the importance of embedding channels, feed-forward neurons, and attention heads using activations from a small calibration dataset. Next, we show that summing layer-wise scores plays a crucial role in performance, whereas the prior work performed ablations along batch and sequence axes. Moreover, for Grouped-Query Attention (GQA)

Models	MMLU *5-shot*	KMMLU *5-shot*	HAERAE *5-shot*	HumanEval *0-shot*	MBPP *3-shot*	GSM8K *5-shot*	Avg
8B†	64.22	48.30	83.41	40.24	51.40	57.09	57.44
4.5B	59.74	48.09	82.58	34.76	48.60	57.01	55.13
2.1B	54.83	44.80	77.09	31.10	46.20	46.32	50.06
1.3B	53.55	39.91	72.59	28.05	39.60	36.01	44.95
635M	46.28	34.60	62.69	23.17	31.40	19.26	36.23
385M	41.16	31.70	47.94	18.90	24.00	10.83	29.08
192M	26.11	30.16	19.71	12.80	12.40	2.43	17.27

Table 6: Performance through iterative compression beyond two iterations. Each model is pruned from the preceding model. † Each model is distilled using the 8B model as the teacher.

(Ainslie et al., 2023), we improve performance by ensuring query-key-value alignment. Specifically, we remove an equal number of query heads within each group, as shown in Figure 9. Additionally, since Kanana employs SwiGLU (Shazeer, 2020), we choose between averaging gate and up states or using intermediate states, whereas the original formulation relies on pre-activation values. All ablation results for importance scoring are in Table 15.

We further enhance the pruning strategies with a focus on intermediate model structures. Consistent with the findings from Minitron, we observe that excessive single-step compression leads to significant degradation. Although maintaining attention heads is generally beneficial, our experiments reveal that pruning them for smaller models is effective when done earlier at larger scales as presented in Table 16. Additionally, we find that input and output embeddings can be tied by averaging without causing noticeable degradation, which we apply when pruning from 4.5B to 2.1B as shown in Table 17.

Lastly, we observe that the composition of distillation data directly influences the performance, while pruning data is less important. For models larger than 2B, we use high-quality 300 billion tokens of stage 2 described in Section 2.3.1. However, for smaller models, increasing the proportion of general-domain English data increases both English performance and other benchmark scores, as shown in Table 18.

In conclusion, our comprehensive pre-training process, which includes staged pre-training, depth up-scaling, and iterative pruning and distillation, offers a compute-efficient strategy for developing high-performing language models. This combined approach not only enhances performance across diverse benchmarks, but also ensures computational efficiency, demonstrating the effectiveness of our strategy in producing a robust family of models spanning the range from 2.1B to 32.5B. See Appendix A.4 for our pruning and distillation configurations.

3 Post-training

Building on Kanana pre-trained models, we further develop instruction-tuned models for direct interaction by natural language. In Section 3.1, we highlight the performance of Kanana instruction-tuned models, demonstrating superior performance on Korean tasks and competitive results on other tasks. Section 3.2 presents the details of the specifics regarding the Supervised Fine-Tuning (SFT) and preference datasets. Section 3.3 outlines the extensive post-training techniques applied on Kanana instruction models.

3.1 Performance

We evaluate our instruction-tuned models across various tasks: chat, instruction following, general knowledge, coding, and mathematics and compare their performance to previous instruction-tuned models. For general chat ability, we use MT-Bench (Zheng et al., 2023), LogicKor (Park, 2024), KoMT-Bench (Research, 2024a), and WildBench (Lin et al., 2025). To

Models	*Chat*				*Instruction Following*
	MT-Bench	LogicKor	KoMT-Bench	WildBench	IFEval
Kanana Flag 32.5B	8.356	**9.524**	**8.058**	54.14	**0.856**
Qwen2.5 32B	8.331	8.988	7.847	51.13	0.822
Gemma 2 27B	8.088	8.869	7.373	46.46	0.817
EXAONE-3.5-32B	**8.375**	9.202	7.907	**54.30**	0.845
Aya Expanse 32B	7.788	8.941	7.626	48.36	0.735
Kanana Essence 9.8B	7.769	8.964	7.706	47.27	0.799
Llama 3.1 8B	7.500	6.512	5.336	33.20	0.772
Qwen2.5 7B	7.625	7.952	6.808	41.31	0.760
Gemma 2 9B	7.633	8.643	7.029	40.92	0.750
EXAONE-3.5-7.8B	**8.213**	**9.357**	**8.013**	**50.98**	**0.826**
Aya Expanse 8B	7.131	8.357	7.006	38.50	0.645
Kanana Nano 2.1B	6.400	7.964	5.857	25.41	0.720
Llama 3.2 3B	7.050	4.452	3.967	21.91	0.767
Qwen2.5 3B	6.969	6.488	5.274	25.76	0.355
Gemma 2 2B	7.225	5.917	4.835	28.71	0.428
EXAONE-3.5-2.4B	**7.919**	**8.941**	**7.223**	**41.68**	**0.790**
Llama 3.1 70B	8.275	8.250	6.970	46.50	0.875
Qwen2.5 72B	8.619	9.214	8.281	55.25	0.861

Table 7: Performance of Kanana and previous instruction-tuned models in general chat and instruction following benchmarks. Across all *Chat* benchmarks, we use `gpt-4o-2024-08-06` as a judge model. The best scores are denoted in **bold**. 70B sized models have been included for reference purposes.

Models	*General*			*Coding*		*Mathematics*	
	MMLU	KMMLU	HAE-RAE	HumanEval+	MBPP+	GSM8K	MATH
Kanana Flag 32.5B	81.08	**64.19**	**68.18**	77.44	69.84	90.83	57.82
Qwen2.5 32B	**84.40**	59.37	48.30	**82.32**	**71.96**	**95.30**	**81.90**
Gemma 2 27B	78.01	49.98	46.02	70.12	70.90	91.05	53.80
EXAONE-3.5-32B	78.30	55.44	52.27	78.66	70.90	93.56	76.80
Aya Expanse 32B	74.49	42.35	51.14	64.63	65.61	75.06	42.82
Kanana Essence 9.8B	70.64	50.76	**47.16**	72.56	69.05	84.91	42.24
Llama 3.1 8B	71.18	39.24	40.91	60.98	57.67	82.71	49.86
Qwen2.5 7B	**77.23**	46.87	37.50	73.78	**70.63**	**91.58**	**75.22**
Gemma 2 9B	73.47	44.47	39.77	59.76	64.55	87.72	48.10
EXAONE-3.5-7.8B	72.62	**52.09**	46.02	**79.27**	66.67	89.99	73.50
Aya Expanse 8B	61.23	35.78	39.20	42.68	56.88	78.85	30.80
Kanana Nano 2.1B	52.48	**38.51**	**33.52**	63.41	62.43	72.32	29.26
Llama 3.2 3B	56.09	3.07	17.05	56.71	50.26	66.57	38.18
Qwen2.5 3B	**69.18**	38.33	32.39	67.68	**64.02**	**84.00**	**65.72**
Gemma 2 2B	57.69	6.99	7.95	35.37	45.24	49.81	21.68
EXAONE-3.5-2.4B	63.19	14.27	14.20	**70.73**	59.79	83.78	64.04
Llama 3.1 70B	83.48	39.08	53.41	75.61	66.40	91.66	63.98
Qwen2.5 72B	87.14	65.78	60.80	81.10	75.66	95.45	82.60

Table 8: Performance of Kanana post-trained models on a set of standard benchmarks. All benchmarks under General category are measured using 0-shot CoT with respective chat-template of each model. The best scores are denoted in **bold**. 70B sized models have been included for reference purposes.

test instruction following ability, we use IFEval[1] (Zhou et al., 2023). For general knowledge tasks, we use MMLU (Hendrycks et al., 2021a), KMMLU (Son et al., 2024a), and HAE-RAE[2] (Son et al., 2024b), with zero-shot chain-of-thought (CoT) (Wei et al., 2022) setting along with the chat template. Employing zero-shot CoT with the chat template, rather than multi-shot prompts, allows us to evaluate the inherent capabilities of the instruction model, without residual traces from the pre-trained model. For coding ability, we use HumanEval+ (Liu et al., 2023) and MBPP+ (Liu et al., 2023). For Mathematical ability, we use GSM8K (Cobbe

[1] We report the average of Prompt-level strict-accuracy and Instruct-level strict-accuracy.

[2] We report general knowledge category scores in this section.

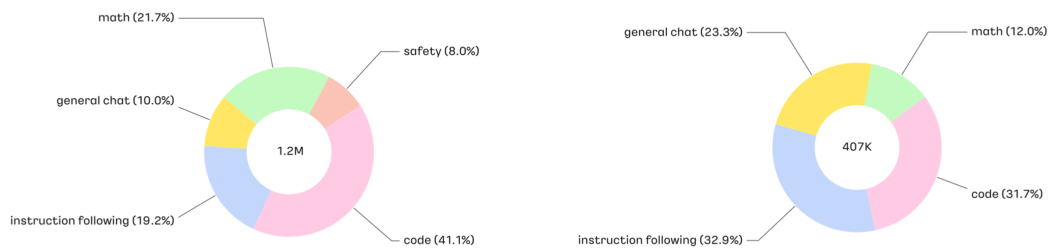
et al., 2021) and MATH (Hendrycks et al., 2021b). See Appendix B.1 for detailed prompts of benchmarks.

Table 7 and Table 8 show that our models excel similar sized models on Korean tasks. The 32.5B model achieves the highest performance in Korean chat tasks (LogicKor, KoMT-Bench) and Korean knowledge tasks (KMMLU, HAE-RAE). The 9.8B and 2.1B models rank second in Korean chat tasks and either best or second-best in Korean knowledge tasks. Additionally, our models exhibit competitive performance across other tasks except in math.

3.2 Data

We collect 1.2M instruction data instances in English and Korean to address both languages. To ensure that our post-training data can handle diverse human requests, we define five distinct domains and collect prompts from both public datasets and human contributors. As a result, our dataset comprises 492K instances for *code*, 260K for *math*, 230K for *instruction following*, 120K for *general chat*, and 96K for *safety*. The safety dataset includes prompts related to ethics, privacy, toxicity, and bias.

Figure 3 depicts the instance size and proportion of each domain. For the preference optimization stage, we sub-sampled and balanced the data across each domain.



(a) SFT data (b) Preference data

Figure 3: Data size and proportion of each domain.

3.3 Training Process

We adopt the widely used multi-stage post-training procedure comprising SFT and a series of preference optimization processes (Ouyang et al., 2022; Grattafiori et al., 2024; Qwen et al., 2025; Team et al., 2024b). In Section 3.3.1, we provide details on the SFT process. In Section 3.3.2, we share information on training our reward model from the SFT model for the subsequent preference optimization process. In Section 3.3.3, we perform preference optimization on the SFT model, which is a sequential process consisting of offline and online preference optimization.

As shown in Figure 4, each step of this process quantitatively enhances the instruction-tuned model across different model sizes. Qualitatively, we observe that during the SFT stage, the model learns to generate structured chat responses while integrating relevant knowledge, and this ability persists through subsequent stages. Building on the SFT model, the preference optimization stages further enhance performance by refining the model's tone and manner. Appendix C presents qualitative results and illustrates the evolution of model completions throughout each phase of post-training.

3.3.1 Supervised Fine-Tuning

During the SFT stage, the model develops the ability to generate structured chat responses while integrating relevant knowledge. In this stage, we train the model using 1.2M data instances, as described in Section 3.2. While optimizing the proportion of domain-specific data, we observed that such data is crucial for achieving high performance in its respective domain and does not negatively impact other domains. Table 9 demonstrates that excluding

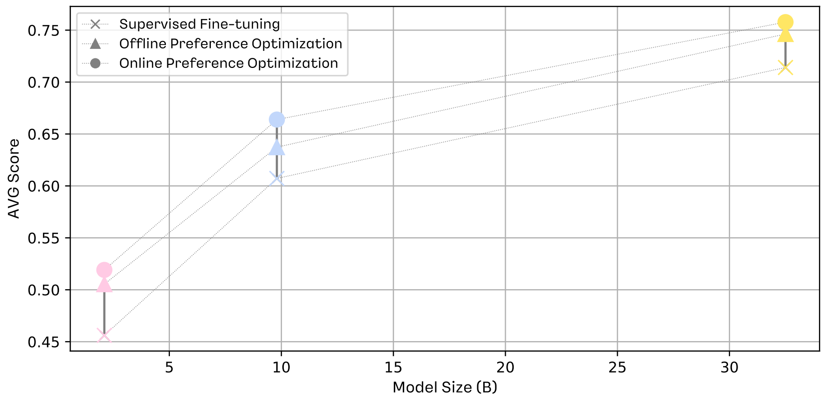


Figure 4: Kanana model performance for each stage of training across different model sizes. The y-axis is the average of normalized scores of all benchmarks in Table 7 and Table 8. The normalization process is done by dividing each score with the maximum possible score.

Datasets used				Normalized Scores					
General	Instruction Following	Code	Math	MT-Bench	IFEval	HumanEval+	MBPP+	GSM8K	MATH
✓	✓	✓	✓	1.00	1.00	1.00	1.00	1.00	1.00
✓	✗	✓	✓	0.98	0.72	1.06	0.99	1.03	1.07
✓	✓	✗	✓	0.99	1.00	0.66	0.72	1.01	1.05
✓	✓	✓	✗	0.98	1.00	1.04	1.00	0.60	0.59

Table 9: Domain mixture ablation for SFT dataset. All scores are normalized by the score of the SFT model when datasets of all domains have been included in the training set. We see that removing a specific domain from the training dataset exclusively deteriorates the performance of the respective domain by a significant amount.

domain-specific data from total dataset only reduces performance on the corresponding domain's benchmark, while performance in other domains remains unaffected. Consequently, we incorporate the full extent of each domain-specific dataset while ensuring balanced performance across all domains.

3.3.2 *Reward Model Training*

We train a reward model for subsequent online preference optimization process, assuming a Bradley-Terry model (Bradley & Terry, 1952). The reward model is trained using the offline preference data along with additional public preference data. Among various reward models trained with different data proportions and settings, we select the one that demonstrates the strongest best-of-N policy (Gao et al., 2023) performance. The best-of-N policy performance is evaluated by generating N responses from the policy model, scoring them with the reward model, selecting the highest-scoring response, and then assessing the final response's quality using a benchmark judge. This approach is based on the intuition that the chosen reward model should effectively evaluate the response distribution of the online preference optimization stage in accordance with the benchmark evaluation criteria.

3.3.3 *Preference Optimization*

To further improve the SFT model's performance on LLM benchmarks, we conduct a preference optimization stage. The process begins with offline preference optimization (Meng et al., 2024; Jung et al., 2024), where we apply direct preference optimization (DPO) (Rafailov et al., 2023) using the offline preference data.

We then conduct online preference optimization, initializing from the offline DPO model. During training, policy-generated responses are evaluated by the reward model from Section 3.3.2, providing training data for online DPO (Guo et al., 2024a) with asynchronous response sampling (Noukhovitch et al., 2025). This approach can be considered as a form of iterative DPO (Xiong et al., 2024). However, unlike prior work (Tran et al., 2023), we maintain a fixed reference model, specifically the offline DPO model, throughout all iterations. This decision is based on our observation that updating the reference model led to undesirable increases in response length.

4 Adaptations

In this section, we show three examples of practical adaptations of Kanana models to popular applications of LLMs: embedding models, retrieval-augmented models, and function calling models. Through experimental results, we show that the performances of Kanana models are further improved in each relevant benchmarks when task-specific training techniques are further applied, showcasing the possibility of adapting Kanana models to a wide range of applications.

4.1 Embedding Models

Text embeddings, or dense vector representations, are essential for capturing the semantic essence of text (Karpukhin et al., 2020; Khattab & Zaharia, 2020). Following the success of LLMs, decoder-only language models have taken their place as a popular backbone of sentence embedding models (Muennighoff, 2022; Wang et al., 2023; Springer et al., 2024; Ma et al., 2024; BehnamGhader et al., 2024; Xu et al., 2024). In this section, we examine the capabilities of the Kanana model, specifically the Kanana Nano 2.1B, as a robust backbone for embedding by employing LLM2Vec (BehnamGhader et al., 2024). For comparative analysis, we also apply LLM2Vec on models of Llama 3 and Qwen2.5 series with similar model sizes.

Embedding Backbone	English	Korean	Avg
Kanana Nano 2.1B	51.56	**65.00**	**58.28**
Llama 3.2 3B	53.28	59.43	56.35
Qwen2.5 3B	**54.00**	62.10	58.05
Llama 3.2 1B	48.77	54.68	51.73
Qwen2.5 1.5B	50.60	54.60	52.60

Table 10: Performance comparison of embedding models on English and Korean retrieval benchmarks. All embedding models are fine-tuned from instruct models. See Appendix D for detailed evaluations.

The embedding models are evaluated on subsets of Massive Text Embedding Benchmark (MTEB) (Muennighoff et al., 2023) retrieval tasks, including 10 English tasks sourced from the MTEB v2 leaderboard (Enevoldsen et al., 2025) and 8 Korean tasks curated by Jang et al. (2024). Table 10 presents average nDCG@10 scores for English and Korean, summarizing the performance results on retrieval tasks.

Kanana Nano 2.1B consistently demonstrates competitive performance and serves as an effective backbone for embedding tasks. As shown in Table 10, our 2.1B model not only significantly surpasses Llama 3.2 1B and Qwen2.5 1.5B across both English and Korean benchmarks, but also outperforms Llama 3.2 3B and Qwen2.5 3B on Korean evaluations, despite its smaller size. Additionally, it achieves a solid English score and the highest average score among the models, highlighting the strong capacity of Kanana Nano 2.1B when fine-tuned for retrieval tasks.

4.2 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) methods (Lewis et al., 2021) enable large language models to access the latest external or proprietary information without altering model parameters (Liu et al., 2024a). In order to ensure factual consistency during retrieval, the grounding ability of the model needs to be trained through additional data mixture (Lin et al., 2024). In this section, we describe a process for developing reliable RAG models with enhanced grounding ability from Kanana LLMs.

For evaluation, we collect RAG scenario benchmarks and evaluate our model on them. ContextualBench (Nguyen et al., 2024) is set of multi-hop QA, which we specifically include to consider the conciseness in evaluation. FACTs (Jacovi et al., 2025) consists of various tasks with contexts such as reasoning, QA, summarization, rewriting, and extraction. [3] IFEval (Zhou et al., 2023) measures maintenance of helpfulness of our instruct model. However, these benchmarks are all English-based, making them insufficient to judge the RAG abilities in Korean. To this end, we develop an internal FACTs-like Korean RAG benchmark called RAG-General-Bench that focuses on measuring factual consistency in Korean. During the development, human annotators manually constructed the dataset with context, instruction, and reference answer, to evaluate helpfulness as well. The benchmark consists of a total of 115 samples with 4 main tasks, categorized into 27 subcategories, providing a diverse set of scenarios for evaluation. There are 2 samples of QA task in Appendix E.

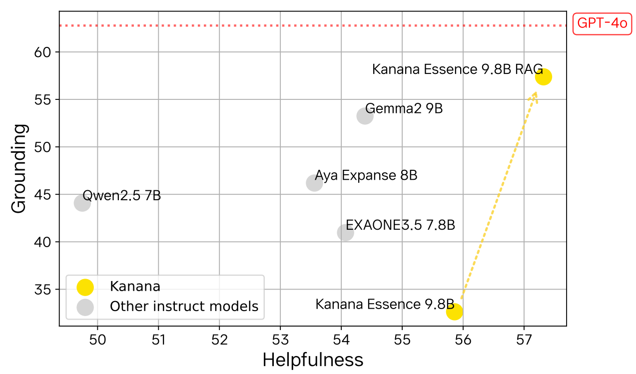


Figure 5: Performance Comparison of Various Models Based on averaged helpfulness and grounding in RAG-General-Bench.

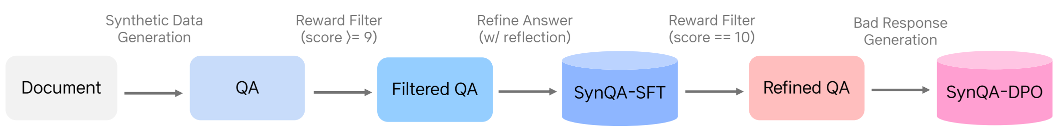


Figure 6: QA Generation Pipeline

To increase grounding ability, we synthetically generate question-answer pairs using high-quality bilingual documents as seed documents, following the pipeline in Figure 6. Then, we filter out instances with low grounding scores and use LLM-judge to reflect and refine the low grounding instances. We call the dataset at this point as SynQA-SFT. With SynQA-SFT, we augment responses with low grounding score to produce preference dataset that we call SynQA-DPO. Along with SynQA datasets, we utilize StructLM (Zhuang et al., 2024) and FollowRAG (Dong et al., 2024) to adapt diverse context format and instructions in RAG scenarios and replay SFT dataset from Section 3.2 to prevent general capability of the instruction model from degrading during training.

However, we observe a decline in the helpfulness score as the model is trained through SFT and DPO in Table 11. In order to address this issue, we merge the DPO model with the instruction model to preserve helpfulness (Kim et al., 2024b). As a result, Kanana Essence 9.8B RAG achieves 91.4% of GPT-4o's grounding performance while maintaining our instruct model's helpfulness in our benchmark as presented in Figure 5.

4.3 Function Calling

Function calling is an essential ability for large language models (LLMs) to interact with external tools and databases, granting them access to up-to-date information stored in

[3] We filtered with character length of 20k since our base model was trained with token length limit of 8k. This dataset is not labeled golden answer, so we only measure grounding score with it.

Models	FACTs Grounding	RAG-General-Bench Grounding	RAG-General-Bench Helpfulness	ContextualBench Exact-match	IFEval
Kanana Essence 9.8B	40.66	32.63	55.86	20.22	**79.93**
+ SFT	62.40	59.29	51.60	48.08	72.99
+ DPO	**63.09**	**65.33**	52.67	**48.76**	75.00
+ Merge (Kanana Essence 9.8B RAG)	53.09	57.38	**57.32**	48.31	78.44

Table 11: Performance change of each phase of recipe. Grounding score is average of two metric RAGAS (Es et al., 2023) Faithfulness and rubric based LLM-judge. Helpfulness score is average of two metric RAGAS Answer Relevancy and rubric based LLM-judge. EM means exact matching normalized answer with golden label. IFEval scoring is as same as Section 3.1.

dynamic or structured formats (Schick et al., 2023). This capability helps integrating real-time data with the static knowledge inherent in LLMs, which is particularly vital in enterprises.

Previous works highlight the increasing importance of function calling, which has led to various efforts in data generation for fine-tuning and model evaluation (Basu et al., 2024; Guo et al., 2024b; Qin et al., 2024; Tang et al., 2023; Li et al., 2023a; Rastogi et al., 2020; Liu et al., 2024b). However, these efforts predominantly focused on English, making it necessary to create a function calling dataset for low-resource languages. To address this gap within Korean contexts, we create a fine-tuning dataset, referred to as `korean-fc-corpus`. The corpus is constructed by: (1) translating two key English function calling corpora, glaive-function-calling-v1 (`gfc-v1`) (GlaiveAI, 2023) and the Schema-Guided Dialogue Dataset (`sgd`) (Rastogi et al., 2020), into their Korean equivalents, `ko-gfc-v1` and `ko-sgd`; and (2) creating an in-house function calling dataset (`inhouse-fc`) specifically tailored for corporate applications.

We further adopt two-staged training process comprising domain specific pre-training and supervised fine-tuning to adapt instruct-tuned models to function calling specific tokens and terminologies. In the domain pre-training phase, we leveraged multiple English-based function calling datasets, including `gfc-v1`, glaive-function-calling-v2 (GlaiveAI, 2024), xlam-function-calling-60k (Liu et al., 2024b), as well as `sgd`, supplemented by our `inhouse-fc`. This foundation enabled us to perform supervised fine-tuning exclusively on `korean-fc-corpus`. This two-stage strategy ensures that models become adequately versed in function calling conventions and domain terminologies before focusing on Korean-specific nuances, thereby enhancing their performance in Korean function calling tasks.

Models	Single-call	Dialogue
Kanana 8B FC	0.88	0.89
gpt-4-0125-preview	0.94	0.94
gpt-4o-2024-05-13	0.93	0.95

Table 12: Evaluation on FunctionChat-Bench: Single-call and Dialogue Accuracy

To evaluate function calling capabilities in corporate environments, we introduce FunctionChat-Bench (Lee et al., 2024), a benchmark designed for Korean conversational settings. This benchmark measures performance on two metrics: Single-call accuracy, which evaluates how well a model selects and invokes the necessary function from several options, and Dialogue accuracy, which examines the model's capability in multi-turn interactions. For comparative analysis, we evaluate OpenAI's proprietary models (gpt-4-0125-preview, and gpt-4o-2024-05-13) and Kanana 8B FC model as shown in Table 12.

This result indicates that leveraging task specific fine-tuning on moderately sized LLMs, which are trained at a lower cost, may offer a more cost effective and efficient approach for addressing certain tasks.

5 Conclusion

In this report, we present Kanana, a family of large language models available in sizes of {2.1B, 9.8B, 32.5B}, with a focus on the cost-effective training procedure compared to other prominent open models. We emphasize the strong bilingual capability of Kanana models, showcasing state-of-the-art performance on Korean benchmarks of KMMLU, HAE-RAE, and KoMT-Bench and competitive results on various English benchmarks. However, we also acknowledge the limitations of Kanana models in overall performance on small scale models sizes, particularly in math domains. To address the limitations, we plan to improve small models and the math ability of all models through data quality and mixture. To further our commitment in cost-effective training, we intend to explore strategical approaches such as formulating scaling laws and other training methodologies as possible future directions. Additionally, we aim to expand the linguistic ability from bilingual to multilingual prioritizing the intuition of treating the underrepresented languages covered in this report. By continuing to build on these efforts, we aspire to make advancements in the field of large language models, balancing performance with efficiency and broadening the linguistic scope of our models.

Contributors and Acknowledgements

Pre-training

Yunju Bak, Doohae Jung, Boseop Kim[†], Nayeon Kim, Hojin Lee, Jaesun Park, Minho Ryu

Post-training

Jiyeon Ham, Seungjae Jung, Hyunho Kim, Hyunwoong Ko, Changmin Lee, Daniel Wontae Nam[†], Kyoung-Woon On[†‡]

Adaptation

Seulye Baeg, Junrae Cho, Taegyeong Eo, Sunghee Jung, Jieun Kang, EungGyun Kim[†], Eunhwa Kim, Byeongil Ko, Daniel Lee, Donghun Lee, Minchul Lee, Miok Lee, Shinbok Lee, Minho Ryu, Gaeun Seo

Acknowledgments

We thank Myungchul Shin and Byung-hak Kim for their invaluable support for Kanana models.

[†]Team leads
[‡]Work done at Kakao Corp.

References

01. AI, :, Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Guoyin Wang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yanpeng Li, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. Yi: Open foundation models by 01.ai, 2025. URL https://arxiv.org/abs/2403.04652.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.298. URL https://aclanthology.org/2023.emnlp-main.298/.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model, 2025. URL https://arxiv.org/abs/2502.02737.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Kinjal Basu, Ibrahim Abdelaziz, Subhajit Chaudhury, Soham Dan, Maxwell Crouse, Asim Munawar, Sadhana Kumaravel, Vinod Muthusamy, Pavan Kapanipathi, and Luis A Lastras. Api-blend: A comprehensive corpora for training and benchmarking api llms. *arXiv preprint arXiv:2402.15491*, 2024.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. Llm2vec: Large language models are secretly powerful text encoders. *arXiv preprint arXiv:2404.05961*, 2024.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023. URL http://jmlr.org/papers/v24/22-1144.html.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, April 2023. URL https://github.com/togethercomputer/RedPajama-Data.

John Dang, Shivalika Singh, Daniel D'souza, Arash Ahmadian, Alejandro Salamanca, Madeline Smith, Aidan Peppin, Sungjin Hong, Manoj Govindassamy, Terrence Zhao, Sandra Kublik, Meor Amer, Viraat Aryabumi, Jon Ander Campos, Yi-Chern Tan, Tom Kocmi, Florian Strub, Nathan Grinsztajn, Yannis Flet-Berliac, Acyr Locatelli, Hangyu Lin, Dwarak Talupuru, Bharat Venkitesh, David Cairuz, Bowen Yang, Tim Chung, Wei-Yin Ko, Sylvie Shang Shi, Amir Shukayev, Sammie Bae, Aleksandra Piktus, Roman Castagné, Felipe Cruz-Salinas, Eddie Kim, Lucas Crawhall-Stein, Adrien Morisot, Sudip Roy, Phil Blunsom, Ivan Zhang, Aidan Gomez, Nick Frosst, Marzieh Fadaee, Beyza Ermis, Ahmet Üstün, and Sara Hooker. Aya expanse: Combining research breakthroughs for a new multilingual frontier, 2024. URL https://arxiv.org/abs/2412.04261.

DeepSeek-AI. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024. URL https://github.com/deepseek-ai/DeepSeek-LLM.

Guanting Dong, Xiaoshuai Song, Yutao Zhu, Runqi Qiao, Zhicheng Dou, and Ji-Rong Wen. Toward general instruction-following alignment for retrieval-augmented generation, 2024. URL https://arxiv.org/abs/2410.09584.

Kenneth Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzemiński, Genta Indra Winata, Saba Sturua, Saiteja Utpala, Mathieu Ciancone, Marion Schaeffer, Gabriel Sequeira, Diganta Misra, Shreeya Dhakal, Jonathan Rystrøm, Roman Solomatin, Ömer Çağatan, Akash Kundu, Martin Bernstorff, Shitao Xiao, Akshita Sukhlecha, Bhavish Pahwa, Rafał Poświata, Kranthi Kiran GV, Shawon Ashraf, Daniel Auras, Björn Plüster, Jan Philipp Harries, Loïc Magne, Isabelle Mohr, Mariya Hendriksen, Dawei Zhu, Hippolyte Gisserot-Boukhlef, Tom Aarsen, Jan Kostkan, Konrad Wojtasik, Taemin Lee, Marek Šuppa, Crystina Zhang, Roberta Rocca, Mohammed Hamdy, Andrianos Michail, John Yang, Manuel Faysse, Aleksei Vatolin, Nandan Thakur, Manan Dey, Dipam Vasani, Pranjal Chitale, Simone Tedeschi, Nguyen Tai, Artem Snegirev, Michael Günther, Mengzhou Xia, Weijia Shi, Xing Han Lù, Jordan Clive, Gayatri Krishnakumar, Anna Maksimova, Silvan Wehrli, Maria Tikhonova, Henil Panchal, Aleksandr Abramov, Malte Ostendorff, Zheng Liu, Simon Clematide, Lester James Miranda, Alena Fenogenova, Guangyu Song, Ruqiya Bin Safi, Wen-Ding Li, Alessia Borghini, Federico Cassano, Hongjin Su, Jimmy Lin, Howard Yen, Lasse Hansen, Sara Hooker, Chenghao Xiao, Vaibhav Adlakha, Orion Weller, Siva Reddy, and Niklas Muennighoff. Mmteb: Massive multilingual text embedding benchmark, 2025. URL https://arxiv.org/abs/2502.13595.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. Ragas: Automated evaluation of retrieval augmented generation, 2023. URL https://arxiv.org/abs/2309.15217.

Maxim Fishman, Brian Chmiel, Ron Banner, and Daniel Soudry. Scaling FP8 training to trillion-token LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=E1EHO0imOb.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10835–10866. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/gao23h.html.

GlaiveAI. glaive-function-calling. https://huggingface.co/datasets/glaiveai/glaive-function-calling, 2023.

GlaiveAI. glaive-function-calling-v2. https://huggingface.co/datasets/glaiveai/glaive-function-calling-v2, 2024.

IBM Granite Team. Granite 3.0 language models, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney

Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary

DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Tom Gunter, Zirui Wang, Chong Wang, Ruoming Pang, Andy Narayanan, Aonan Zhang, Bowen Zhang, Chen Chen, Chung-Cheng Chiu, David Qiu, Deepak Gopinath, Dian Ang Yap, Dong Yin, Feng Nan, Floris Weers, Guoli Yin, Haoshuo Huang, Jianyu Wang, Jiarui Lu, John Peebles, Ke Ye, Mark Lee, Nan Du, Qibin Chen, Quentin Keunebroek, Sam Wiseman, Syd Evans, Tao Lei, Vivek Rathod, Xiang Kong, Xianzhi Du, Yanghao Li, Yongqiang Wang, Yuan Gao, Zaid Ahmed, Zhaoyang Xu, Zhiyun Lu, Al Rashid, Albin Madappally Jose, Alec Doane, Alfredo Bencomo, Allison Vanderby, Andrew Hansen, Ankur Jain, Anupama Mann Anupama, Areeba Kamal, Bugu Wu, Carolina Brum, Charlie Maalouf, Chinguun Erdenebileg, Chris Dulhanty, Dominik Moritz, Doug Kang, Eduardo Jimenez, Evan Ladd, Fangping Shi, Felix Bai, Frank Chu, Fred Hohman, Hadas Kotek, Hannah Gillis Coleman, Jane Li, Jeffrey Bigham, Jeffery Cao, Jeff Lai, Jessica Cheung, Jiulong Shan, Joe Zhou, John Li, Jun Qin, Karanjeet Singh, Karla Vega, Kelvin Zou, Laura Heckman, Lauren Gardiner, Margit Bowler, Maria Cordell, Meng Cao, Nicole Hay, Nilesh Shahdadpuri, Otto Godwin, Pranay Dighe, Pushyami Rachapudi, Ramsey Tantawi, Roman Frigg, Sam Davarnia, Sanskruti Shah, Saptarshi Guha, Sasha Sirovica, Shen Ma, Shuang Ma, Simon Wang, Sulgi Kim, Suma Jayaram, Vaishaal Shankar, Varsha Paidi, Vivek Kumar, Xin Wang, Xin Zheng, Walker Cheng, Yael Shrager, Yang Ye, Yasu Tanaka, Yihao Guo, Yunsong Meng, Zhao Tang Luo, Zhi Ouyang, Alp Aygar, Alvin Wan, Andrew Walkingshaw, Andy Narayanan, Antonie Lin, Arsalan Farooq, Brent Ramerth, Colorado Reed, Chris Bartels, Chris Chaney, David Riazati, Eric Liang Yang, Erin Feldman, Gabriel Hochstrasser, Guillaume Seguin, Irina Belousova, Joris Pelemans, Karen Yang, Keivan Alizadeh Vahid, Liangliang Cao, Mahyar Najibi, Marco Zuliani, Max Horton, Minsik Cho, Nikhil Bhendawade, Patrick Dong, Piotr Maj, Pulkit Agrawal, Qi Shan, Qichen Fu, Regan Poston, Sam Xu, Shuangning Liu, Sushma Rao, Tashweena Heeramun, Thomas Merth, Uday Rayala, Victor Cui, Vivek Rangarajan Sridhar, Wencong Zhang, Wenqi Zhang, Wentao Wu, Xingyu Zhou, Xinwen Liu, Yang Zhao, Yin Xia, Zhile Ren, and Zhongzheng Ren. Apple intelligence foundation language models, 2024. URL https://arxiv.org/abs/2407.21075.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, Johan Ferret, and Mathieu Blondel. Direct language model alignment from online ai feedback, 2024a. URL https://arxiv.org/abs/2402.04792.

Zhen Guo, Adriana Meza Soria, Wei Sun, Yikang Shen, and Rameswar Panda. Api pack: A massive multi-programming language dataset for api call generation. *arXiv preprint arXiv:2402.09615*, 2024b.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021b.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack W. Rae, and Laurent Sifre. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Shengding Hu, Yuge Tu, Xu Han, Ganqu Cui, Chaoqun He, Weilin Zhao, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Xinrong Zhang, Zhen Leng Thai, Chongyi

Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, dahai li, Zhiyuan Liu, and Maosong Sun. MiniCPM: Unveiling the potential of small language models with scalable training strategies. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=3X2L2TFr0f.

Siming Huang, Tianhao Cheng, J. K. Liu, Jiaran Hao, Liuyihan Song, Yang Xu, J. Yang, J. H. Liu, Chenchen Zhang, Linzheng Chai, Ruifeng Yuan, Zhaoxiang Zhang, Jie Fu, Qian Liu, Ge Zhang, Zili Wang, Yuan Qi, Yinghui Xu, and Wei Chu. Opencoder: The open cookbook for top-tier code large language models, 2024. URL https://arxiv.org/abs/2411.04905.

Adam Ibrahim, Benjamin Thérien, Kshitij Gupta, Mats Leon Richter, Quentin Gregory Anthony, Eugene Belilovsky, Timothée Lesort, and Irina Rish. Simple and scalable strategies to continually pre-train large language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=DimPeeCxKO.

INF-Team. Inf's open-source large language models. 2024. URL https://s.infly.cn/f/img/pdf/inf_34b_tech_report.pdf.

Alon Jacovi, Andrew Wang, Chris Alberti, Connie Tao, Jon Lipovetz, Kate Olszewska, Lukas Haas, Michelle Liu, Nate Keating, Adam Bloniarz, Carl Saroufim, Corey Fry, Dror Marcus, Doron Kukliansky, Gaurav Singh Tomar, James Swirhun, Jinwei Xing, Lily Wang, Madhu Gurumurthy, Michael Aaron, Moran Ambar, Rachana Fellinger, Rui Wang, Zizhao Zhang, Sasha Goldshtein, and Dipanjan Das. The facts grounding leaderboard: Benchmarking llms' ability to ground responses to long-form input, 2025. URL https://arxiv.org/abs/2501.03200.

Youngjoon Jang, Junyoung Son, and Taemin Lee. KURE: Korea university retrieval embedding model, 2024. URL https://github.com/nlpai-lab/KURE.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In Mirella Lapata, Phil Blunsom, and Alexander Koller (eds.), *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 427–431, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-2068/.

Seungjae Jung, Gunsoo Han, Daniel Wontae Nam, and Kyoung-Woon On. Binary classifier optimization for large language model alignment, 2024. URL https://arxiv.org/abs/2404.04656.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 39–48, 2020.

Sanghoon Kim, Dahyun Kim, Chanjun Park, 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. In Yi Yang, Aida Davani, Avi Sil, and Anoop Kumar (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, pp. 23–35, Mexico City, Mexico, June 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-industry.3. URL https://aclanthology.org/2024.naacl-industry.3/.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models, 2024b. URL https://arxiv.org/abs/2405.01535.

Solomon Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951. URL https://api.semanticscholar.org/CorpusID:120349231.

Shinbok Lee, Gaeun Seo, Daniel Lee, Byeongil Ko, Sunghee Jung, and Myeongcheol Shin. Functionchat-bench: Comprehensive evaluation of language models' generative capabilities in korean tool-use dialogs. *arXiv preprint arXiv:2411.14054*, 2024.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021. URL https://arxiv.org/abs/2005.11401.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Kumar Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee F Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Kamal Mohamed Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Joshua P Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah M Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham M. Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander T Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alex Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. Datacomp-LM: In search of the next generation of training sets for language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=CNWdWn47IE.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. API-bank: A comprehensive benchmark for tool-augmented LLMs. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3102–3116, Singapore, December 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.187. URL https://aclanthology.org/2023.emnlp-main.187/.

Raymond Li, Loubna Ben allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia LI, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue Zhuo, Thomas Wang, Olivier Dehaene, Joel Lamy-Poirier, Joao Monteiro, Nicolas Gontier, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Ben Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy, Jason T Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Urvashi Bhattacharyya, Wenhao Yu, Sasha Luccioni, Paulo Villegas, Fedor Zhdanov, Tony Lee, Nadav Timor, Jennifer Ding, Claire S Schlesinger, Hailey Schoelkopf, Jan Ebert, Tri Dao, Mayank Mishra, Alex Gu, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro Von Werra, and Harm de Vries. Starcoder: may the source be with you! *Transactions on Machine Learning Research*, 2023b. ISSN 2835-8856. URL https://openreview.net/forum?id=KoFOg41haE. Reproducibility Certification.

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. Wildbench: Benchmarking LLMs with challenging tasks from real users in the wild. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=MKEHCx25xp.

Sheng-Chieh Lin, Luyu Gao, Barlas Oguz, Wenhan Xiong, Jimmy Lin, Wen tau Yih, and Xilun Chen. Flame: Factuality-aware alignment for large language models, 2024. URL https://arxiv.org/abs/2405.01525.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatGPT really correct? rigorous evaluation of large language models for code generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=1qvx610Cu7.

Zihan Liu, Wei Ping, Rajarshi Roy, Peng Xu, Chankyu Lee, Mohammad Shoeybi, and Bryan Catanzaro. Chatqa: Surpassing gpt-4 on conversational qa and rag, 2024a. URL https://arxiv.org/abs/2401.10225.

Zuxin Liu, Thai Quoc Hoang, Jianguo Zhang, Ming Zhu, Tian Lan, Shirley Kokane, Juntao Tan, Weiran Yao, Zhiwei Liu, Yihao Feng, Rithesh R N, Liangwei Yang, Silvio Savarese, Juan Carlos Niebles, Huan Wang, Shelby Heinecke, and Caiming Xiong. APIGen: Automated PIpeline for generating verifiable and diverse function-calling datasets. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024b. URL https://openreview.net/forum?id=Jfg3vw2bjx.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, Tianyang Liu, Max Tian, Denis Kocetkov, Arthur Zucker, Younes Belkada, Zijian Wang, Qian Liu, Dmitry Abulkhanov, Indraneil Paul, Zhuang Li, Wen-Ding Li, Megan Risdal, Jia Li, Jian Zhu, Terry Yue Zhuo, Evgenii Zheltonozhskii, Nii Osae Osae Dade, Wenhao Yu, Lucas Krauß, Naman Jain, Yixuan Su, Xuanli He, Manan Dey, Edoardo Abati, Yekun Chai, Niklas Muennighoff, Xiangru Tang, Muhtasham Oblokulov, Christopher Akiki, Marc Marone, Chenghao Mou, Mayank Mishra, Alex Gu, Binyuan Hui, Tri Dao, Armel Zebaze, Olivier Dehaene, Nicolas Patry, Canwen Xu, Julian McAuley, Han Hu, Torsten Scholak, Sebastien Paquet, Jennifer Robinson, Carolyn Jane Anderson, Nicolas Chapados, Mostofa Patwary, Nima Tajbakhsh, Yacine Jernite, Carlos Muñoz Ferrandis, Lingming Zhang, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. Starcoder 2 and the stack v2: The next generation, 2024. URL https://arxiv.org/abs/2402.19173.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2421–2425, 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=3Tzcot1LKb.

AI Meta. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. *Meta AI Blog. Retrieved December*, 20:2024, 2024.

Niklas Muennighoff. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*, 2022.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In Andreas Vlachos and Isabelle Augenstein (eds.), *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2014–2037, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.eacl-main.148. URL https://aclanthology.org/2023.eacl-main.148/.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Bhuminand Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Xuan-Phi Nguyen, Shrey Pandit, Senthil Purushwalkam, Austin Xu, Hailin Chen, Yifei Ming, Zixuan Ke, Silvio Savarese, Caiming Xong, and Shafiq Joty. Sfr-rag: Towards contextually faithful llms, 2024. URL https://arxiv.org/abs/2409.09916.

Michael Noukhovitch, Shengyi Huang, Sophie Xhonneux, Arian Hosseini, Rishabh Agarwal, and Aaron Courville. Faster, more efficient RLHF through off-policy asynchronous learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=FhTAG591Ve.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Jeonghwan Park. Logickor. Available at https://github.com/instructkr/LogicKor, 2024.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024a. URL https://openreview.net/forum?id=n6SCkn2QaG.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Martin Jaggi, Leandro von Werra, and Thomas Wolf. Fineweb2: A sparkling update with 1000s of languages, December 2024b. URL https://huggingface.co/datasets/HuggingFaceFW/fineweb-2.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=HPuSIXJaa9.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 8689–8696, 2020.

LG AI Research. KoMT-bench. https://huggingface.co/datasets/LGAI-EXAONE/KoMT-Bench, 2024a.

LG AI Research. Exaone 3.5: Series of large language models for real-world use cases. *arXiv preprint arXiv:2412.04862*, 2024b.

Noveen Sachdeva, Benjamin Coleman, Wang-Cheng Kang, Jianmo Ni, Lichan Hong, Ed Huai hsin Chi, James Caverlee, Julian J. McAuley, and Derek Zhiyuan Cheng. How to train data-efficient llms. *ArXiv*, abs/2402.09668, 2024. URL https://api.semanticscholar.org/CorpusID:267682083.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=Yacmpz84TH.

Zhihong Shao, Damai Dai, Daya Guo, Bo Liu (Benjamin Liu), Zihan Wang, and Huajian Xin. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *ArXiv*, abs/2405.04434, 2024a. URL https://api.semanticscholar.org/CorpusID:269613809.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024b. URL https://arxiv.org/abs/2402.03300.

Noam Shazeer. GLU variants improve transformer. *CoRR*, abs/2002.05202, 2020. URL https://arxiv.org/abs/2002.05202.

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. *CoRR*, abs/2402.11548, 2024a. URL https://doi.org/10.48550/arXiv.2402.11548.

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 Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 7993–8007, Torino, Italia, May 2024b. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.704/.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*, 2024.

Sharath Turuvekere Sreenivas, Saurav Muralidharan, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Llm pruning and distillation in practice: The minitron approach. *arXiv preprint arXiv:2408.11796*, 2024.

Dan Su, Kezhi Kong, Ying Lin, Joseph Jennings, Brandon Norick, Markus Kliegl, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-cc: Transforming common crawl into a refined long-horizon pretraining dataset. *arXiv preprint arXiv:2412.02595*, 2024.

Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, Boxi Cao, and Le Sun. Toolalpaca: Generalized tool learning for language models with 3000 simulated cases. *arXiv preprint arXiv:2306.05301*, 2023.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff

Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. Gemma: Open models based on gemini research and technology, 2024a. URL https://arxiv.org/abs/2403.08295.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024b. URL https://arxiv.org/abs/2408.00118.

Hoang Tran, Chris Glaze, and Braden Hancock. Iterative dpo alignment. Technical report, Snorkel AI, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Alan Wake, Bei Chen, C. X. Lv, Chao Li, Chengen Huang, Chenglin Cai, Chujie Zheng, Daniel Cooper, Fan Zhou, Feng Hu, Ge Zhang, Guoyin Wang, Heng Ji, Howard Qiu, Jiangcheng Zhu, Jun Tian, Katherine Su, Lihuan Zhang, Liying Li, Ming Song, Mou Li, Peng Liu, Qicheng Hu, Shawn Wang, Shijun Zhou, Shiming Yang, Shiyong Li, Tianhang Zhu, Wen Xie, Wenhao Huang, Xiang He, Xiaobo Chen, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Yanpeng Li, Yongke Zhao, Yongzhen Luo, Yuchi Xu, Yuxuan Sha, Zhaodong Yan, Zhiyuan Liu, Zirui Zhang, and Zonghong Dai. Yi-lightning technical report, 2025. URL https://arxiv.org/abs/2412.01253.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*, 2023.

Ruizhe Wang, Yeyun Gong, Xiao Liu, Guoshuai Zhao, Ziyue Yang, Baining Guo, Zhengjun Zha, and Peng Cheng. Optimizing large language model training using fp4 quantization, 2025. URL https://arxiv.org/abs/2501.17116.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Mitchell Wortsman, Peter J Liu, Lechao Xiao, Katie E Everett, Alexander A Alemi, Ben Adlam, John D Co-Reyes, Izzeddin Gur, Abhishek Kumar, Roman Novak, Jeffrey Pennington, Jascha Sohl-Dickstein, Kelvin Xu, Jaehoon Lee, Justin Gilmer, and Simon Kornblith. Small-scale proxies for large-scale transformer training instabilities. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=d8w0pmvXbZ.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for RLHF under KL-constraint. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54715–54754. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/xiong24a.html.

Ran Xu, Wenqi Shi, Yue Yu, Yuchen Zhuang, Yanqiao Zhu, May Dongmei Wang, Joyce C. Ho, Chao Zhang, and Carl Yang. BMRetriever: Tuning large language models as better biomedical text retrievers. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22234–22254, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1241. URL https://aclanthology.org/2024.emnlp-main.1241/.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. GaLore: Memory-efficient LLM training by gradient low-rank projection. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61121–61143. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhao24s.html.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=uccHPGDlao.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023. URL https://arxiv.org/abs/2311.07911.

Alex Zhuang, Ge Zhang, Tianyu Zheng, Xinrun Du, Junjie Wang, Weiming Ren, Wenhao Huang, Jie Fu, Xiang Yue, and Wenhu Chen. StructLM: Towards building generalist models for structured knowledge grounding. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=EKBPn7no4y.

A Appendix

A.1 Comparison between pre-trained models and post-trained models

Models	Tokens	Category	MMLU 5-shot	KMMLU 5-shot	HAE-RAE 5-shot	HumanEval 0-shot	MBPP 3-shot	GSM8K 5-shot
Kanana Flag 32.5B	3.2T	base	77.68	62.10	90.47	51.22	63.40	70.05
		instruct	77.84	62.08	89.37	64.63	73.00	84.08
Llama 3.1 70B	15T	base	78.93	53.00	76.35	57.32	66.60	81.73
		instruct	82.42	52.80	76.08	78.05	70.40	86.66
Qwen2.5 32B	18T	base	83.10	63.15	75.16	50.00	73.40	82.41
		instruct	83.41	61.20	74.61	54.88	73.00	76.27
Gemma 2 27B	13T	base	75.45	51.16	69.11	51.22	64.60	74.37
		instruct	76.39	51.49	68.84	71.34	66.20	84.46
EXAONE-3.5-32B	6.5T	instruct	72.68	46.36	82.22	74.39	67.80	55.50
Aya Expanse 32B	-	instruct	74.52	49.57	80.66	12.20	60.40	85.97
Kanana Essence 9.8B	3.2T	base	67.61	50.57	84.97	40.24	53.60	63.61
		instruct	66.45	49.95	82.95	61.59	51.60	76.04
Llama 3.1 8B	15T	base	65.18	41.02	61.78	35.37	48.60	50.87
		instruct	68.17	41.22	64.44	59.76	58.00	69.52
Qwen2.5 7B	18T	base	74.19	51.68	67.46	56.71	63.20	83.85
		instruct	74.23	50.13	65.72	65.85	31.60	77.56
Gemma 2 9B	8T†	base	70.34	48.18	66.18	37.20	53.60	68.16
		instruct	72.30	46.56	66.73	56.10	57.60	80.12
EXAONE-3.5-7.8B	9T	instruct	65.36	45.30	77.54	70.73	61.60	64.67
Aya Expanse 8B	-	instruct	62.52	40.11	71.95	7.93	47.40	75.97
Kanana Nano 2.1B	300B†	base	54.83	44.80	77.09	31.10	46.20	46.32
		instruct	53.67	42.92	77.17	54.88	55.00	64.37
Llama 3.2 3B	9T†‡	base	56.40	35.57	47.66	25.61	39.00	27.37
		instruct	60.60	35.44	48.21	49.39	49.00	58.76
Qwen2.5 3B	18T	base	65.57	45.28	61.32	37.80	55.60	69.07
		instruct	66.47	44.51	60.77	50.61	54.60	11.37
Gemma 2 2B	2T†	base	52.89	30.67	45.55	20.12	28.20	24.72
		instruct	57.04	33.48	49.77	23.78	37.80	44.05
EXAONE-3.5-2.4B	6.5T	instruct	59.27	43.58	68.65	63.41	58.40	53.07

Table 13: † For distilled models, distillation tokens are only counted ‡ Information from https://huggingface.co/meta-llama/Llama-3.2-3B

A.2 Suboptimal extraction of open-source datasets

Example of suboptimal extraction from arXiv

```
(...)
In this work, we use sine and cosine functions of different frequencies:

\begin{align*}
    PE_{(pos,2i)} = sin(pos / 10000^{2i/d_{\text{model}}}) \\
    PE_{(pos,2i+1)} = cos(pos / 10000^{2i/d_{\text{model}}})
\end{align*}

where $pos$ is the position and $i$ is the dimension.  That is, each
dimension of the positional encoding corresponds to a sinusoid.  The
wavelengths form a geometric progression from $2\pi$ to $10000 \cdot 2\
pi$.  We chose this function because we hypothesized it would allow the
model to easily learn to attend by relative positions, since for any
fixed offset $k$, $PE_{pos+k}$ can be represented as a linear function of
 $PE_{pos}$.

We also experimented with using learned positional embeddings \citep{
JonasFaceNet2017} instead, and found that the two versions produced
nearly identical results (see Table~\ref{tab:variations} row (E)).  We
chose the sinusoidal version because it may allow the model to
extrapolate to sequence lengths longer than the ones encountered during
training.

\section{Introduction}

\input{introduction}

\section{Background}

\input{background}

\section{Model Architecture}
\input{model_architecture}

\section{Why Self-Attention}
\input{why_self_attention}

\section{Training}
\input{training}

\section{Results} \label{sec:results}
\input{results}

\section{Conclusion}
In this work, we presented the Transformer, the first sequence
transduction model based entirely on attention, replacing the recurrent
layers most commonly used in encoder-decoder architectures with multi-
headed self-attention.
(...)
```

Figure 7: Example of suboptimal extraction from arXiv subset of Computer (2023). The original content is from Vaswani et al. (2017).

Example of suboptimal extraction from Wikipedia

발생원인

회전 좌표계

회전좌표계 좌표계 x, y, z와 좌표계 x′, y′, z′을 보자 두 좌표계의 원점은 같다. 각각의 경우에 대해 벡터 .은 두 좌표계에서 다음과 같이 표시된다.

. (x, y, z 좌표)

. (x′, y′, z′ 좌표계)

벡터의 내적을 이용해 x, y, z를 (.), (.), (.)으로 표현할 수 있다. 내적의 방법은 다음과 같다.

.

.

.
으로 표현되는 것을 확인할 수 있다.

(a) Open-source

Example of improved extraction from Wikipedia

발생원인

회전 좌표계

회전좌표계
좌표계 x, y, z와 좌표계 x′, y′, z′을 보자 두 좌표계의 원점은 같다. 각각의 경우에 대해 벡터 **r**.은 두 좌표계에서 다음과 같이 표시된다.

$\mathbf{r} = x\hat{x} + y\hat{y} + z\hat{z}$. (x, y, z 좌표)

$\mathbf{r} = x'\hat{x}' + y'\hat{y}' + z'\hat{z}'$. (x′, y′, z′ 좌표계)

벡터의 내적을 이용해 x, y, z를 ($x', \hat{x}', \hat{x}$.), ($y', \hat{y}', \hat{y}$.), ($z', \hat{z}', \hat{z}$.)으로 표현할 수 있다. 내적의 방법은 다음과 같다.

$\mathbf{r}\hat{x} = x = (x'\hat{x}' + y'\hat{y}' + z'\hat{z}')(\hat{x}) = x'(\hat{x}'\hat{x}) + y'(\hat{y}'\hat{x}) + z'(\hat{z}'\hat{x})$.

$\mathbf{r}\hat{y} = y = (x'\hat{x}' + y'\hat{y}' + z'\hat{z}')(\hat{y}) = x'(\hat{x}'\hat{y}) + y'(\hat{y}'\hat{y}) + z'(\hat{z}'\hat{y})$.

$\mathbf{r}\hat{z} = z = (x'\hat{x}' + y'\hat{y}' + z'\hat{z}')(\hat{z}) = x'(\hat{x}'\hat{z}) + y'(\hat{y}'\hat{z}) + z'(\hat{z}'\hat{z})$.

으로 표현되는 것을 확인할 수 있다.

(b) Improved

Figure 8: Example of suboptimal and our improved extraction from open-source Wikipedia dataset (https://huggingface.co/datasets/wikimedia/wikipedia). The original content is from the Korean Wikipedia article on the Coriolis effect.

A.3 Details of pre-training from scratch

To control the effects of architecture and tokenization, and to focus on improving the data scaling curve, we adopt the architecture and tokenizer of Llama 3 (Grattafiori et al., 2024). Note that while we use the Llama 3 tokenizer, we do not utilize either the weights or the outputs of Llama 3 during the training of Kanana. Based on the observations of Wortsman et al. (2024), we adopt independent weight decay, which follows the original proposal of Loshchilov & Hutter (2019) and differs from the PyTorch implementation, and a z-loss (Chowdhery et al., 2023) to obtain effective and stable training across various model scales. We set an independent weight decay of 1×10^{-4} and a z-loss coefficient of 5×10^{-6}, regardless of model size. For peak learning rates, learning rate schedulers, and batch sizes, the hyperparameter scaling law and multi-step scheduler from DeepSeek-AI (2024) are employed.

A.4 Details of Pruning and Distillation

The hyperparameters differ from those used in pre-training from scratch. We apply a cosine learning rate schedule (Loshchilov & Hutter, 2017) with an initial learning rate of 1.2×10^{-4}, batch size of 512, sequence length of 8192, and a warmup phase of 100 steps. Following the recommendation of Minitron (Muralidharan et al., 2024; Sreenivas et al., 2024), we employ KL divergence (Kullback & Leibler, 1951) on final logits as the sole loss function. Additionally, we conclude training early during ablation studies, as pruned models quickly regain performance and the ranking of ablation options rapidly stabilizes.

Models	**MMLU** *5-shot*	**KMMLU** *5-shot*	**HAE-RAE** *5-shot*	**HumanEval** *0-shot*	**MBPP** *3-shot*	**GSM8K** *5-shot*	**Avg**
Kanana 4.5B	59.74	48.09	82.58	34.76	48.60	57.01	55.13
Kanana 3B	58.21	**47.55**	**79.19**	34.15	45.90	53.75	53.13
Llama 3.2 3B	56.40	35.57	47.66	25.61	39	27.37	38.60
Qwen2.5 3B	**65.57**	45.28	61.32	**37.80**	**55.60**	**69.07**	**55.77**
Kanana 2.1B	54.83	**44.80**	**77.09**	31.10	**46.20**	46.32	**50.06**
Kanana 1.3B	53.55	39.91	72.59	28.05	39.60	36.01	44.95
Gemma 2 2B	52.89	30.67	45.55	20.12	28.20	24.72	33.69
SmolLM2-1.7B	50.08	24.36	30.52	0.61	34.00	32.00	28.60
Qwen2.5 1.5B	**60.86**	36.63	49.68	**37.20**	44.00	62.09	48.41
Llama 3.2 1B	31.51	26.46	23.10	18.90	27.60	6.14	22.29
Kanana 635M	46.28	**34.60**	**62.69**	23.17	**31.40**	19.26	**36.23**
Kanana 385M	41.16	31.70	47.94	18.90	24.00	10.83	29.09
Kanana 192M	26.11	30.16	19.71	12.80	12.40	2.43	17.27
Qwen2.5 0.5B	**47.59**	31.79	31.44	**28.66**	31.00	**35.10**	34.26
SmolLM2-360M	24.84	15.14	21.26	0.00	19.00	3.94	14.03
SmolLM2-135M	25.28	25.73	20.71	0.00	3.40	1.29	12.74

Table 14: Performance of our models obtained with iterative pruning & distillation, compared to similar-sized open-source base models.

GQA alignment	**Swiglu importance**	**Aggregation** Layer	Batch	Sequence	**Avg**
✓	**intermediate states**	**sum**	**l2norm**	**avg**	**36.41**
✗	intermediate states	sum	l2norm	avg	20.13
✓	avg of gate, up states	sum	l2norm	avg	36.04
✓	intermediate states	✗	l2norm	avg	13.81
✓	intermediate states	sum	avg	avg	35.65
✓	intermediate states	sum	l2norm	l2norm	34.25

Table 15: Ablation study on importance scoring details, followed by training the same 1.3B architecture with 25B tokens.

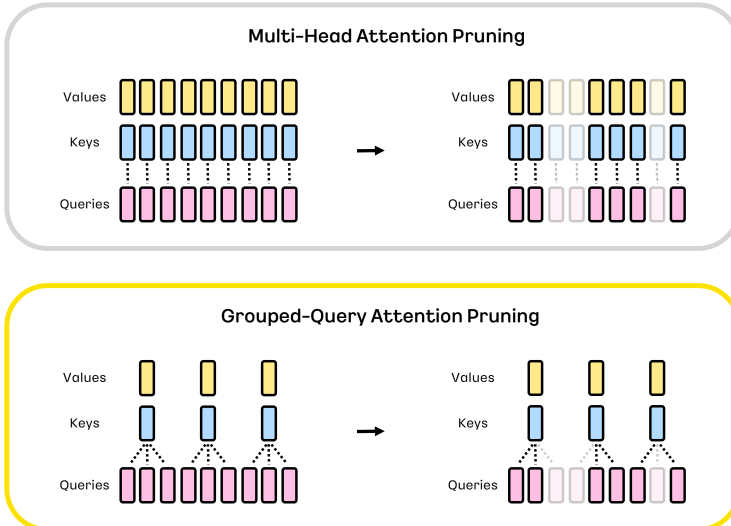


Figure 9: Illustration of ensuring query-key-value alignment in GQA pruning.

Hidden	Intermediate	Query heads	Non-embedding parameters	Avg
1280	5120	**24**	0.96B	**32.81**
1280	5760	16	0.96B	28.71
1280	5760	**24**	1.04B	**34.27**
1536	4608	16	0.98B	31.99
1536	4608	**24**	1.08B	**35.10**
1536	5376	8	0.99B	24.39
1536	5376	16	1.09B	32.39
1536	6144	8	1.11B	25.91
1024	3072	24→16	1.08B→504M	20.85
1024	3072	**16→16**	1.09B→504M	**21.78**

Table 16: Ablation study on model architectures, using 25B training tokens.

Embedding	MMLU *5-shot*	KMMLU *5-shot*	HAERAE *5-shot*	HumanEval *0-shot*	MBPP *3-shot*	GSM8K *5-shot*	Avg
tied	49.07	40.41	70.49	30.49	40.60	38.21	44.88
untied	49.88	39.61	70.21	29.88	40.20	36.92	44.45

Table 17: Ablation study on tying input and output embeddings by averaging, using 63B training tokens. The rest of the architecture remains unchanged, with 1.86B non-embedding parameters.

Models	Data	MMLU *5-shot*	KMMLU *5-shot*	HAERAE *5-shot*	HumanEval *0-shot*	MBPP *3-shot*	GSM8K *5-shot*	Avg
1.3B	stage2	39.52	26.65	49.77	**25.00**	32.40	**21.00**	32.39
1.3B	**stage2 en++**	**44.00**	**33.90**	**62.42**	23.78	**33.80**	20.55	**36.41**

Table 18: Ablation study on distillation data, using 25B training tokens.

MMLU prompt (0-shot CoT)

The following are multiple choice questions about {mmlu_subject}. Summarize your reasoning concisely, then conclude with "Therefore, the answer is: X" where X is one of A, B, C, or D.

Question: {question}
A. {choice_A}
B. {choice_B}
C. {choice_C}
D. {choice_D}

(a) MMLU prompt

KMMLU prompt (0-shot CoT)

다음은 {kmmlu_subject}에 관한 객관식 문제입니다. 당신의 추론 과정을 간결하게 요약한 후, "따라서, 정답은: X"라고 결론지으십시오. 여기서 X는 A, B, C, D 중 하나입니다.

질문: {question}
A. {choice_A}
B. {choice_B}
C. {choice_C}
D. {choice_D}

(b) KMMLU prompt

HAE-RAE (0-shot CoT)

다음은 객관식 문제입니다. 당신의 추론 과정을 간결하게 요약한 후, "따라서, 정답은: X"라고 결론지으십시오. 여기서 X는 A, B, C, D, E 중 하나입니다.
{query}

(c) HAE-RAE prompt

Figure 10: Evaluation prompts for MMLU, KMMLU, and HAE-RAE. The prompts are used to evaluate instruction-tuned models.

B Evaluation Details

B.1 Evaluation Prompts for post-trained models

We employ 0-shot CoT prompts for a number of evaluations. See Figure 10 for MMLU, KMMLU and HAE-RAE. For math-related tasks we employ 0-shot prompt. We refer the readers to Figure 11 for the prompts corresponding to GSM8K and MATH.

GSM8K (0-shot)

Put your final answer within \boxed{}.

{question}

(a) GSM8K prompt

MATH (0-shot)

Put your final answer within \boxed{}.

{problem}

(b) MATH prompt

Figure 11: Evaluation prompts for GSM8K and MATH. The prompts are used to evaluate instruction-tuned models.

C Qualititive Results

In this section, we include qualitative experimental results. We mainly focus on changes made during post-training. See Table 19 and Table 20 for representative examples.

Prompt	**1 더하기 1은 귀요미야! 1 더하기 1은?**
SFT	1 더하기 1은 2입니다.
DPO	1 더하기 1은 2입니다. 귀요미라는 말은 상황에 따라 다르게 해석될 수 있지만, 일반적으로는 '귀여운 것'을 의미합니다. 예를 들어, 아이들이 수학 문제를 재미있게 풀기 위해 "1 더하기 1은 귀요미" 같은 구호를 사용하기도 합니다. 하지만 수학적으로는 1 + 1 = 2입니다.
Online DPO	1 더하기 1은 2입니다. 귀요미라는 표현은 수학적인 계산보다는 감정적인 표현으로 보입니다. 혹시 다른 질문이나 추가적인 설명이 필요하시면 알려주세요!

Table 19: Prompt and Completions of each post-training stage.

Prompt	**Convert these dates to YYYY/MM/DD format:** **12/31/2021** **31-12-21** **01/02/2022** **02-01-22**

SFT	Here is a Python solution for the problem:
	<pre>def convert_date(date):
	    if '-' in date:
	        date_parts = date.split('-')
	        return f"{date_parts[2]}/{date_parts[1]}/{date_parts[0]}"
	    else:
	        date_parts = date.split('/')
	        return f"{date_parts[2]}/{date_parts[0]}/{date_parts[1]}"
	
	dates = ["12/31/2021", "31-12-21", "01/02/2022", "02-01-22"]
	for date in dates:
	    print(convert_date(date))</pre>When you run this code, it will print:
	
	2021/12/31
	2021/12/31
	2022/02/01
	2022/01/02
	This code works by checking if the input date contains a '-' character. If it does, it assumes the date is in the format DD-MM-YYYY and converts it to YYYY/MM/DD. If the input date does not contain a '-' character, it assumes the date is in the format MM/DD/YYYY and converts it to YYYY/MM/DD.

Continued on next page

Table 20 – *Continued from previous page*

Prompt	**Convert these dates to YYYY/MM/DD format:** **12/31/2021** **31-12-21** **01/02/2022** **02-01-22**

DPO	Here is a Python solution for the problem:
	`def convert_date(date):`
	`    if '/' in date:`
	`        month, day, year = date.split('/')`
	`        return f"{year}/{month.zfill(2)}/{day.zfill(2)}"`
	`    elif '-' in date:`
	`        day, month, year = date.split('-')`
	`        return f"{year}/{month.zfill(2)}/{day.zfill(2)}"`
	`    else:`
	`        raise ValueError("Invalid date format")`
	
	`dates = ["12/31/2021", "31-12-21", "01/02/2022", "02-01-22"]`
	
	`for date in dates:`
	`    print(convert_date(date))`
	This script will convert the dates to the desired format. The zfill(2) function is used to add leading zeros if the month or day is a single digit.
	
	When you run this script, it will output:
	
	2021/12/31
	2021/12/31
	2022/01/02
	2022/01/02
	Note that the dates "31-12-21" and "02-01-22" are interpreted as "31 December 2021" and "2 January 2022" respectively, because the year is given as "21" and "22". If these dates are meant to represent "12-31-2021" and "1-2-2022" (i.e., December 31, 2021 and January 2, 2022), then the script will need to be modified to handle this interpretation.

Online DPO	Sure, here are the dates converted to the YYYY/MM/DD format: 12/31/2021 → 2021/12/31 31-12-21 → 2021/12/31 01/02/2022 → 2022/01/02 02-01-22 → 2022/01/02 Note that the dates "31-12-21" and "02-01-22" are ambiguous and could potentially mean the opposite year, but based on common sense and the context provided, the most likely interpretation is the one given above (i.e., 2021 and 2022 respectively). If the exact year is unknown, it's crucial to have additional context.

Table 20: Prompt and Completions of each post-training stage.

D Evaluation Details of Embedding Models

Benchmarks	Kanana-Nano-2.1B	Llama3.2 3B	Qwen2.5 3B	Llama3.2 1B	Qwen2.5 1.5B
English (Avg)	51.56	53.28	54.00	48.77	50.60
ArguAna	54.59	54.36	56.26	51.80	53.49
CQADupstackGamingRetrieval	58.37	60.31	59.65	56.13	57.20
CQADupstackUnixRetrieval	43.34	45.27	45.41	39.18	41.14
ClimateFEVERHardNegatives	29.64	30.64	31.08	26.93	27.66
FEVERHardNegatives	73.18	79.09	80.26	73.27	72.09
FiQA2018	40.22	46.47	47.12	38.54	41.08
HotpotQAHardNegatives	61.35	66.10	66.33	61.21	64.18
SCIDOCS	21.41	21.44	22.14	18.96	19.81
TRECCOVID	79.85	81.84	80.87	72.67	75.88
Touche2020Retrieval.v3	53.63	47.26	50.91	49.00	53.50
Korean (Avg)	65.00	59.43	62.10	54.68	54.60
AutoRAGRetrieval	79.71	70.87	75.64	71.47	72.32
BelebeleRetrieval	92.35	87.58	90.16	84.44	83.53
Ko-StrategyQA	79.98	73.92	76.38	63.46	64.97
MIRACLRetrieval	60.04	52.25	56.83	48.28	48.68
MrTidyRetrieval	49.82	45.83	48.48	35.32	37.94
MultiLongDocRetrieval	30.17	25.54	25.75	20.98	17.13
PublicHealthQA	88.08	84.12	86.68	80.26	79.71
XPQARetrieval	39.88	35.33	36.89	33.24	32.55

Table 21: Evaluation details of embedding models on English and Korean retrieval benchmarks.

E RAG-General-Bench Examples

context:

{doc1}
□1 마스크 착용 명령 대상
❶의무화 장소 · 시설 · 대상 ◯ (마스크 착용 의무)
− 감염취약시설 중 입소형 시설, 의료기관 · 약국 및 대중교통수단의 실내*
* 실내란 버스 · 택시 · 기차 · 선박 · 항공기, 기타 차량 등 운송수단, 건축물 및 사방이 구획되어 외부와 분리되어 있는 모든 구조물을 의미
■감염취약시설 중 입소형 시설
* (제외장소) 사무동 · 기숙사 등 입소자의 출입이 필요 없는 구역(단, 건물 또는 층 단위로 구역이 구분되는 경우만 해당)
(...)

{doc2}
□2 마스크착용의무위반시과태료
❶대상별 과태료 금액
◯마스크 착용 위반 당사자(행정명령 위반 당사자): 위반 횟수와 관계없이 각각 10만원 과태료 부과(법 제83조제4항)
(...)

instruction:

A씨는 최근에 요양병원에 계신 어머님을 두 번 면회하고 왔습니다.
A씨는 두 번 다 마스크를 착용하지 않았지만 별다른 제지는 없었습니다.
A씨와 요양병원 원장 B씨가 각각 내야 할 과태료는 얼마인가요?

reference_answer:

요양병원은 마스크 착용 의무화된 곳으로 규정되어 있습니다. 따라서 A씨는 두 번 다 마스크를 착용하지 않았으므로, 위반 당사자인 A씨는 위반 횟수와 관계 없이 각각 10만원의 과태료를 부과받으므로, 총 20만원의 과태료를 내야 합니다. 요양병원 원장 B씨는 관리자로서 이용자에게 마스크 착용을 안내하고 강제하는 의무가 있습니다. 만약 B씨가 해당 의무를 다하지 못했다면, 과태료의 부과 기준에 따라 1차 위반 시 50만원, 2차 위반 시 100만원의 과태료를 부과받게 됩니다. 따라서 B씨는 총 150만원의 과태료를 내야합니다.

(a) Sample 1

context:

{doc1}
2023년 교육부 소관 비영리법인 현황
[{"연번":"1","법인명":"한국인문사회총연합회","소관부서":"학술연구정책과"},
(...)
{"연번":"24","법인명":"소비자교육중앙회","소관부서":"디지털소통팀"}]

{doc2}
2024년 교육부 소관 비영리법인 현황
[{"연번":"1","법인명":"한국인문사회총연합회","소관부서":"학술연구정책과"},
(...)
{"연번":"24","법인명":"소비자교육중앙회","소관부서":"학부모정책과"}]

instruction:

교육부 소관 비영리법인 중 2023년과 비교해서 2024년에 소관부서가 달라진 법인을 '법인명, 2023년 소관부서, 2024년 소관부서'로 구분해서 표 형태로 보여주세요.

reference_answer:

2023년과 비교해서 2024년에 소관부서가 달라진 법인들은 다음과 같습니다.

```
| 법인명 | 2023년 소관부서 | 2024년 소관부서 |
|---------------------------------|---------------------------------------|----------------------------------|
| 한국대학법인협의회 | 대학경영지원과 | 대학경영혁신지원과 |
| 참교육을위한전국학부모회 | 디지털소통팀 | 학부모정책과 |
| 소비자교육중앙회 | 디지털소통팀 | 학부모정책과 |
```

(b) Sample 2

Figure 12: RAG-General-Bench Example: QA

F FunctionChat-Bench Examples

F.1 Single-call

Single-call evaluates how accurately the LM can select and call the necessary function among several options by providing four single-turn prompts for each of 25 different functions. As

context:

{doc1}
□1 마스크 착용 명령 대상
❶의무화 장소 · 시설 · 대상 ○ (마스크 착용 의무)
– 감염취약시설 중 입소형 시설, 의료기관 · 약국 및 대중교통수단의 실내*
* 실내란 버스 · 택시 · 기차 · 선박 · 항공기, 기타 차량 등 운송수단, 건축물 및 사방이 구획되어 외부와 분리되어 있는 모든 구조물을 의미
■감염취약시설 중 입소형 시설
* (제외장소) 사무동 · 기숙사 등 입소자의 출입이 필요 없는 구역(단, 건물 또는 층 단위로 구역이 구분되는 경우만 해당)
(...)

{doc2}
□2 마스크착용의무위반시과태료
❶대상별 과태료 금액
○마스크 착용 위반 당사자(행정명령 위반 당사자): 위반 횟수와 관계없이 각각 10만원 과태료 부과(법 제83조제4항)
(...)

instruction:

A씨는 최근에 요양병원에 계신 어머님을 두 번 면회하고 왔습니다.
A씨는 두 번 다 마스크를 착용하지 않았지만 별다른 제지는 없었습니다.
A씨와 요양병원 원장 B씨가 각각 내야 할 과태료는 얼마인가요?

reference_answer:

요양병원은 마스크 착용 의무화된 곳으로 규정되어 있습니다. 따라서 A씨는 두 번 다 마스크를 착용하지 않았으므로, 위반 당사자인 A씨는 위반 횟수와 관계 없이 각각 10만원의 과태료를 부과받으므로, 총 20만원의 과태료를 내야 합니다. 요양병원 원장 B씨는 관리자로서 이용자에게 마스크 착용을 안내하고 강제하는 의무가 있습니다. 만약 B씨가 해당 의무를 다하지 못했다면, 과태료의 부과 기준에 따라 1차 위반 시 50만원, 2차 위반 시 100만원의 과태료를 부과받게 됩니다. 따라서 B씨는 총 150만원의 과태료를 내야합니다.

(a) Sample 1

context:

{doc1}
2023년 교육부 소관 비영리법인 현황
[{"연번":"1","법인명":"한국인문사회총연합회","소관부서":"학술연구정책과"},
(...)
{"연번":"24","법인명":"소비자교육중앙회","소관부서":"디지털소통팀"}]

{doc2}
2024년 교육부 소관 비영리법인 현황
[{"연번":"1","법인명":"한국인문사회총연합회","소관부서":"학술연구정책과"},
(...)
{"연번":"24","법인명":"소비자교육중앙회","소관부서":"학부모정책과"}]

instruction:

교육부 소관 비영리법인 중 2023년과 비교해서 2024년에 소관부서가 달라진 법인을 '법인명, 2023년 소관부서, 2024년 소관부서'로 구분해서 표 형태로 보여주세요.

reference_answer:

2023년과 비교해서 2024년에 소관부서가 달라진 법인들은 다음과 같습니다.

법인명	2023년 소관부서	2024년 소관부서
한국대학법인협의회	대학경영지원과	대학경영혁신지원과
참교육을위한전국학부모회	디지털소통팀	학부모정책과
소비자교육중앙회	디지털소통팀	학부모정책과

(b) Sample 2

Figure 13: RAG-General-Bench Example: QA

show in Figure 14, "1_exact" is that only the target function is provided to the Assistant as a candidate.

Tools:

```
{
    "type": "function",
    "function": {
      "name": "getDailyNewsHeadlines",
      "description": "오늘의 주요 뉴스 헤드라인을 제공합니다.",
      "parameters": {
        "type": "object",
        "properties": {},
        "required": []
      }
    }
}
```

Context:
user: 오늘 뉴스 알려줄 수 있어?

Assistant:

```
{
  "type": "function",
  "function": {
    "name": " getDailyNewsHeadlines ",
    "arguments": "{}"
  }
}
```

Figure 14: FunctionChat-bench Example : Single-call(1_exact)

F.2 Dialogue

The dialog dataset consists of 45 diverse multi-turn interactions between real users and an LM, categorized into four situation types to evaluate the model's response accuracy and appropriateness.

1. **Call**: An LM must accurately select functions and extract the necessary parameters to respond to a user prompt
2. **Completion**: An LM must generate appropriate responses based on the results of the tool.
3. **Slot**: An LM must query the user for the necessary parameters to make a function call.
4. **Relevance**: An LM must generate an appropriate response when it cannot provide a function for a user prompt.

Tools:
{ "type": "function", ...}

Context:
user: 제리 출국날이 언제였지?

Assistant:
tool_calls {"type": "function", "function": {"name": "informDday", "arguments": "{\"searchTerm\": \"제리 출국날\"}"}}

[Call]

Tools:
{ "type": "function", ...}

Context:
user: 제리 출국날이 언제였지?
assistant: tool_calls {"type": "function", "function": {"name": "informDday", "arguments": "{\"searchTerm\": \"제리 출국날\"}"}}
tool: {"name": "informDday", "content": "{\"ddayName\": \"제리 출국날\", \"ddayDate\": \"2024-04-23\", \"daysRemaining\": 48, \"daysSince\": None}"}

Assistant:
제리 출국날은 2024년 4월 23일입니다. 앞으로 48일 남았습니다.

[Completion]

Tools:
{ "type": "function", ...}

Context:
user: 제리 출국날이 언제였지?
assistant: tool_calls {"type": "function", "function": {"name": "informDday", "arguments": "{\"searchTerm\": \"제리 출국날\"}"}}
tool: {"name": "informDday", "content": "{\"ddayName\": \"제리 출국날\", \"ddayDate\": \"2024-04-23\", \"daysRemaining\": 48, \"daysSince\": None}"} assistant: 제리 출국날은 2024년 4월 23일입니다. 앞으로 48일 남았습니다.
user: 송별회 일정 잡기 to do list에 추가해줘.

Assistant:
알겠습니다. 데드라인이 언제인가요?

[Slot]

Tools:
{ "type": "function", ...}

Context:
user: 송별회 일정 잡기 to do list에 추가해줘.
assistant: 알겠습니다. 데드라인이 언제인가요?
user: 다음주 토요일
assistant: tool_calls {"type": "function", "function": {"name": "add_task", "arguments": "{\"task_name\": \"송별회 일정 잡기\", \"deadline\": \"다음주 토요일\"}"}}
tool: {"name": "add_task", "content": "{\"status\": \"success\"}"}
assistant: 할 일을 등록했습니다
user: 다빈이한테 괜찮을 때 전화 한번 달라고 문자 남겨줘.

Assistant:
문자 전송 기능은 없습니다.

[Relevance]

Figure 15: FunctionChat-bench Example : Dialogue