dots.llm1 Technical Report

rednote-hilab

🤗 https://huggingface.co/rednote-hilab

○ https://github.com/rednote-hilab/dots.llm1

Abstract

Mixture of Experts (MoE) models have emerged as a promising paradigm for scaling language models efficiently by activating only a subset of parameters for each input token. In this report, we present dots.llm1, a large-scale MoE model that activates 14 billion parameters out of a total of 142 billion parameters, delivering performance on par with state-of-the-art models while reducing training and inference costs. Leveraging our meticulously crafted and efficient data processing pipeline, dots.llm1 achieves performance comparable to Qwen2.5-72B after pretraining on 11.2T high-quality tokens and post-training to fully unlock its capabilities. Notably, no synthetic data is used during pretraining. To foster further research, we open-source intermediate training checkpoints at every one trillion tokens, providing valuable insights into the learning dynamics of large language models.
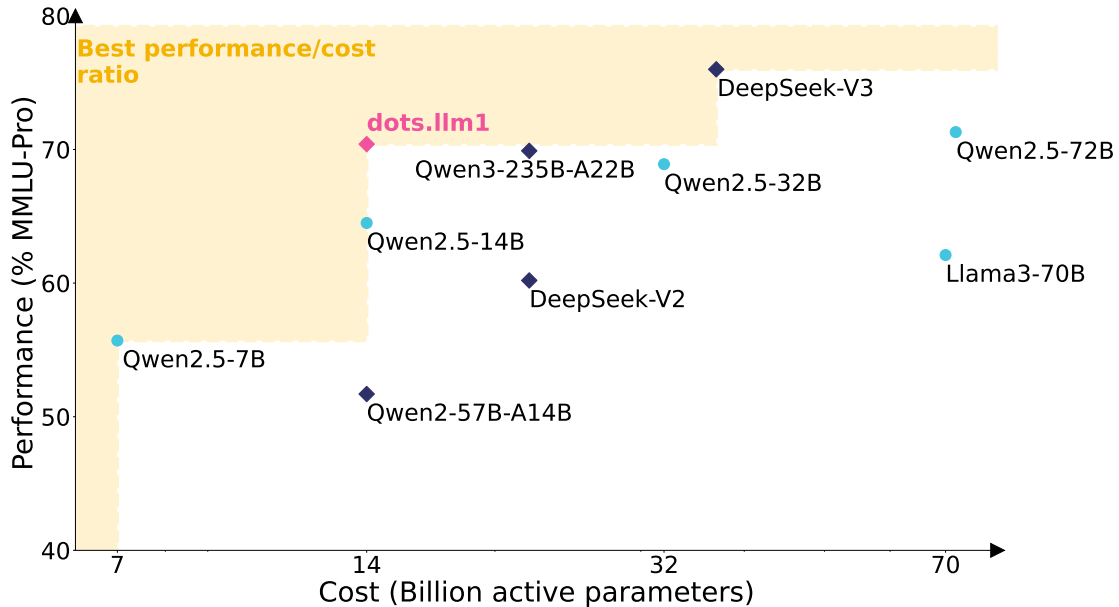
Figure 1: Performance and cost comparison of open MoE and dense language models. Circles (○) denote dense models, while diamonds (◇) denote MoE models. We benchmark model capabilities using MMLU-Pro, showing that dots.llm1 achieves comparable accuracy to leading models.

1 Introduction

Large Language Models (LLMs) have undergone rapid advancements in recent years, moving closer to the goal of Artificial General Intelligence (AGI) as evidenced by substantial progress (OpenAI, 2025a;b; Anthropic, 2025; xAI, 2025). Parallel to these proprietary developments, the open-source community is also achieving remarkable breakthroughs (Qwen, 2024a; DeepSeek-AI et al., 2024; Mistral, 2025; AI@Meta, 2024b). These initiatives are making substantial efforts to close the performance gap with closed-source models, driving forward the evolution of LLMs in the open-source domain.

Mixture of Experts (MoE) (Fedus et al., 2022) is a neural network architecture that divides the model into multiple expert networks, each specializing in different aspects of the input data. By dynamically routing input tokens to a subset of these experts, MoE achieves both computational efficiency and scalability. In recent years, MoE has been widely adopted in the development of large language models (LLMs) (Jiang et al., 2024; Qwen, 2024b; DeepSeek-AI et al., 2024), enabling them to scale to large-scale model sizes while maintaining efficient resource utilization. This approach has proven instrumental in pushing the boundaries of LLM performance, as it allows for the activation of only a fraction of the model's parameters during inference, significantly reducing computational overhead. Beyond LLMs, MoE has also found extensive applications in other domains (Riquelme et al., 2021; Shi et al., 2024; Jawaid et al., 2024), which showcases its versatility and effectiveness.

In this paper, we introduce dots.llm1, a powerful and cost-effective mixture-of-experts model with 142B parameters, of which 14B are activated for each token. dots.llm1 achieves efficient inference on a single node equipped with eight GPUs (40GB or 80GB memory), delivering performance on par with leading open-source models across a wide array of tasks.

The dots.llm1 model adopts a sparse DeepSeekMoE framework (Dai et al., 2024), employing a classic multi-head attention (MHA) (Vaswani et al., 2017) mechanism combined with QK-Norm to ensure training stability. Additionally, it incorporates the innovative auxiliary-loss-free strategy (Wang et al., 2024a; DeepSeek-AI et al., 2024) to effectively manage load balancing, thereby minimizing any potential negative impacts of the balancing process on model performance. On the data side, we introduce a three-stage data processing framework to address the pressures of both growing data volume and high-quality requirements through document preparation, rule-based processing, and model-based processing. On the infrastructure side, we introduce an innovative MoE all-to-all communication and computation overlapping recipe based on 1F1B pipeline scheduling and an efficient grouped GEMM implementation to boost computational efficiency.

During the pre-training phase, dots.llm1 utilizes 11.2 trillion diverse, high-quality tokens. Following this, the context length is further extended from 8K to 32K. In the post-training phase, the model undergoes efficient supervised fine-tuning (SFT) using 400K carefully curated instruction-tuning instances. dots.llm1 is evaluated on a suite of pre-training and post-training benchmarks and compared with state-of-the-art models. It demonstrates balanced, robust performance across multiple domains, with notable strengths in Chinese language processing and mathematical reasoning.

Lastly, we will open-source intermediate training checkpoints at every one trillion tokens, aiming to push the boundaries of LLM development and empower the broader community to build efficient and powerful language models.

We summarize our main contributions as follows:

- **Enhanced Data Processing**: We propose a scalable and fine-grained three-stage data processing framework designed to generate large-scale, high-quality and diverse data for pretraining. The complete process is openly provided to facilitate reproducibility.

- **Performance and Cost Efficiency**: We introduce dots.llm1, an open-source model that activates only 14B parameters during inference while delivering comprehensive and computationally efficient performance. Trained on 11.2 trillion high-quality tokens generated through our scalable data processing framework, dots.llm1 demonstrates robust performance across diverse tasks, all achieved without reliance on synthetic data or model distillation.

- **Infrastructure**: we introduce an innovative MoE all-to-all communication and computation overlapping recipe based on 1F1B pipeline scheduling and an efficient grouped GEMM implementation to boost computational efficiency.

- **Open Accessibility to Model Dynamics**: By releasing intermediate training checkpoints as open-source, we aim to empower the research community with transparency into the training process, enabling deeper insights into the dynamics of large models and fostering accelerated innovation in the field of LLM.

2 Architecture

The `dots.llm1` model is a decoder-only Transformer architecture (Vaswani et al., 2017), where each layer consists of an attention layer and a Feed-forward Network(FFN). Unlike dense models such as Llama (AI@Meta, 2024a) or Qwen (Yang et al., 2024), the FFN is replaced with a Mixture of Experts (MoE) module (Jiang et al., 2024; DeepSeek-AI, 2024a;b; DeepSeek-AI et al., 2024). This modification allows for the training of highly capable models while maintaining an economical cost.

Attention Layer   We utilize a vanilla multi-head attention mechanism (Vaswani et al., 2017) in our model. Following Dehghani et al. (2023), RMSNorm is applied to the query and key projections prior to computing attention. This normalization mitigates the risk of excessively large attention logits, which could otherwise destabilize the training process (Wortsman et al., 2023; Tian et al., 2024a; OLMo et al., 2024).

Mixture-of-Experts Layer   Following DeepSeek-AI (2024a); Qwen (2024b), we replace the FFN with a Mixture-of-Experts (MoE) layer comprising both shared and isolated experts. Our implementation features 128 routed experts and 2 shared experts activated for all tokens, with each expert implemented as a fine-grained, two-layer FFN utilizing `SwiGLU` (Shazeer, 2020) activation. For each token, the router selects the top-6 isolated experts in addition to the 2 shared experts, resulting in 8 active experts per token. Notably, we employ FP32 precision for the gating layer computations rather than BF16 to ensure numerical stability and more accurate expert selection during the routing process.

Load Balancing   Imbalanced expert loading in the MoE layer can lead to routing collapse, diminishing both model capacity and computational efficiency during training and inference. To address this issue, we adopt an auxiliary-loss-free approach (Wang et al., 2024a), as also employed in DeepSeek-AI et al. (2024). It introduces a bias term for each expert, which is added to the corresponding affinity scores to determine the top-k routing. This bias term is dynamically adjusted during training to maintain a balanced load across experts. In addition, we also employ a sequence-wise balance loss to prevent extreme imbalance within any single sequence. Due to the effective load balancing strategy, `dots.llm1` maintains good load balance throughout training and does not drop any tokens during training.

3 Infrastructures

The training of `dots.llm1` is underpinned by the internal Cybertron framework, which is a lightweight training framework built upon Megatron-Core. Leveraging Megatron-Core, we have meticulously constructed a comprehensive suite of toolkits for both model pre-training and post-training. For distinct training stages, namely pre-training, supervised fine-tuning (SFT), and reinforcement learning (RL), we have encapsulated separate trainers in a unified manner to guarantee the coherence and high efficiency of the training process.

3.1 Interleaved `1F1B` based Communication and Computation Overlap

We propose an innovative interleaved `1F1B` based all-to-all communication and computation overlap solution (NVIDIA, 2024b), and work with NVIDIA to integrate it into Megatron-Core. Compared to the vanilla interleaved `1F1B` pipeline, we add an additional step to the warm-up phase, which incurs neither additional bubble overheads nor additional GPU memory consumption for activations. Under this refined pipeline scheduling scheme, during the steady `1F1B` phase, all-to-all communication and computation within the forward and backward step pairs can be effectively overlapped, similar to the DualPipe proposed by DeepSeek (DeepSeek-AI et al., 2024). Compared with the DualPipe, our approach demonstrates a notable advantage in terms of memory consumption, albeit with a marginally higher bubble rate.

3.2 Efficient Implementation of Grouped GEMM

Grouped GEMMs play a significant role in the computation of MoE architectures. A straightforward approach is to schedule sub-GEMM problems in a unified manner across streaming multiprocessors. Some frameworks pre-compute the mapping between tiles and thread blocks on the host side to reduce scheduling overhead on the device (Li et al., 2019). Other works have explored scheduling strategies for small or variable-sized batched GEMMs (Yang et al., 2022; Zhang et al., 2022). Additionally, NVIDIA has introduced Grouped GEMM APIs in cuBLAS (Hejazi, 2024) and Transformer Engine (TE) (NVIDIA, 2024a).
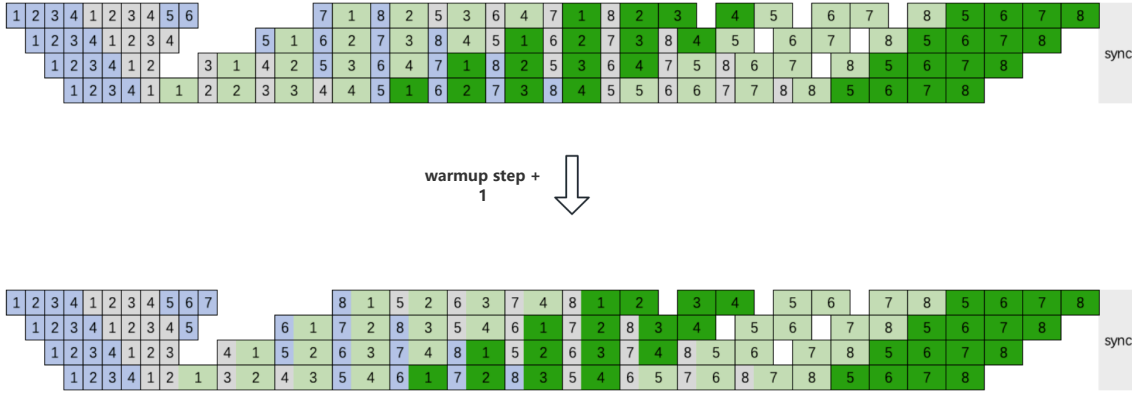
Figure 2: Interleaved 1F1B based communication and computation overlap

Table 1: Performance comparison of Transformer Engine 2.1 and our implementation on H800. Here, g, m, n, and k represent the number of groups (experts) and the dimensions of each sub-GEMM problem, respectively.

	(g, m, n, k)	TE's (TFLOPS)	Ours (TFLOPS)	Speed-up
forward	(4, 1024, 2816, 4096)	589.12	662.48	12.45%
	(4, 1024, 4096, 2816)	581.77	725.35	24.68%
	(4, 2048, 2816, 4096)	636.49	678.89	6.66%
	(4, 2048, 4096, 2816)	636.94	745.45	17.04%
	(8, 1024, 2816, 4096)	614.87	654.77	6.49%
	(8, 1024, 4096, 2816)	609.69	745.01	22.19%
	(8, 2048, 2816, 4096)	646.70	705.85	9.15%
	(8, 2048, 4096, 2816)	657.20	744.74	13.32%
backward	(4, 1024, 2816, 4096)	614.24	665.06	8.27%
	(4, 1024, 4096, 2816)	594.03	637.48	7.31%
	(4, 2048, 2816, 4096)	655.39	709.75	8.29%
	(4, 2048, 4096, 2816)	635.02	677.51	6.69%
	(8, 1024, 2816, 4096)	633.35	677.56	6.98%
	(8, 1024, 4096, 2816)	619.97	637.71	2.86%
	(8, 2048, 2816, 4096)	666.00	706.85	6.13%
	(8, 2048, 4096, 2816)	650.53	695.38	6.89%

To allow large tile sizes and reduce scheduling overhead, we align M_i (token segment of expert i) to a fixed block size. This fixed block size must be a multiple of M in tile shape modifier mMnNkK of the asynchronous warpgroup level matrix multiply-accumulate (WGMMA) (wgmma.mma_async) instruction. Thus, the warpgroups in a single threadblock will have a unified tiling, and the whole token segment (M_i) processed by a threadblock necessarily belongs to the same expert, making the scheduling very similar to that of normal GEMMs.

Compared to NVIDIA's Grouped GEMM APIs in Transformer Engine (v2.1), our implementation exhibits notable advantages. Table 1 presents a performance comparison for forward and backward computations on the H800, where tokens are evenly routed to experts. Our approach achieves an average performance improvement of 14.00% for forward computation and 6.68% for backward computation.

4 Pre-training

4.1 Pre-training Data

The quantity, quality, and diversity of training data play a crucial role in determining the performance of language models. To achieve these objectives, we have structured our approach into three key stages: document preparation, rule-based processing, and model-based processing. This design allows us to efficiently manage large volumes of data, even with limited computational resources. Document preparation focuses on preprocessing and organizing the raw data. Rule-based processing aims to

minimize the need for extensive human curation by automatically filtering and cleaning the data. Model-based processing further ensures that the final dataset is both high-quality and diverse. To ensure safety, we applied filters to exclude data from websites likely to contain unsafe content or significant amounts of personally identifiable information (PII), as well as domains flagged as harmful by a safety classifier. During pre-training, we maintain a balanced 1:1 ratio of Chinese to English data, and no synthetic data is used.

The data processing pipeline is thoroughly documented in Appendix C. Two key innovations distinguish our data processing pipeline as follows:

Web Clutter Removal Model   To address issues such as boilerplate content and repetitive lines, we develop a lightweight model that operates at the line level. This approach achieves an effective balance between cleaning quality and computational efficiency, representing a unique feature that is not commonly found in open-source datasets.

Category Balancing   We train a 200-class classifier to balance the proportions within the web data. This enables us to increase the presence of knowledge-based and factual content, such as encyclopedia entries and popular science articles, while reducing the share of fictional and highly structured web content, including science fiction novels and product descriptions.

4.2   Hyper-Parameters

The dots.llm1 model is trained using the AdamW optimizer (Loshchilov & Hutter, 2017), with $\beta_1 = 0.9$ and $\beta_2 = 0.95$. We implement a weight decay of 0.1 and apply gradient clipping at 1.0. Following Dai et al. (2024); DeepSeek-AI (2024b); DeepSeek-AI et al. (2024), the weights are initialized from a normal distribution with standard deviation 0.006. dots.llm1 comprises 62 layers, with the first layer utilizing a vanilla dense FFN and the subsequent layers employing MoE.

We set the maximum sequence length to 8K during pre-training and train dots.llm1 on 11.2T tokens. Following Hu et al. (2024), we utilize a warmup-stable-decay learning rate schedule. The learning rate warms up over 4,000 steps before stabilizing at 3×10^{-4} for the stable training phase, which encompasses 10T tokens of data. We progressively increase the batch size from 64M tokens initially to 96M tokens at 6T tokens, and finally to 128M tokens at 8.3T tokens.

After the main training phase, the process includes two annealing stages, comprising a total of 1.2 trillion tokens of data.

Stage 1: We train for 1T tokens while gradually decreasing the learning rate from 3×10^{-4} to 3×10^{-5}. During this annealing stage, we significantly increase the proportion of reasoning-related and knowledge-related data to 90%.

Stage 2: We continue training for 200B tokens while reducing the learning rate from 3×10^{-5} to 1×10^{-5} and increasing the proportion of code, mathematics, and reasoning data.

4.3   Long Context Extension

We implement context length extension after the annealing phases. During this phase, we maintain a constant learning rate while training on 128B tokens using UtK strategy (Tian et al., 2024b), extending the sequence length to 32K. Rather than modifying the datasets, UtK tries chunking of training documents into smaller segments, then training the model to reconstruct relevant segments from shuffled chunks. By learning to untie these knotted chunks, the model can effectively handle longer input sequences while maintaining its performance on short-context tasks.

4.4   Evaluation

4.4.1   Benchmarks

To comprehensively evaluate dots.llm1 model, which is pretrained on both Chinese and English, we assess its performance across a suite of benchmarks spanning multiple domains in each language. All evaluations are conducted using the vLLM framework (Kwon et al., 2023). We categorize the benchmarks as follows:

Language Understanding   To evaluate English contextual reasoning and reading comprehension, we employ benchmarks such as HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), the ARC suite (Clark et al., 2018), BigBenchHard (BBH) (Suzgun et al., 2023), and DROP (Dua et al., 2019).

Table 2: Comparison between `dots.llm1` and other representative open-source base models. All models were evaluated under identical conditions. Results demonstrate that `dots.llm1` achieves superior performance compared to DeepSeek-V2, while delivering results comparable to Qwen2.5 72B.

	Benchmark (Metric)	# Shots	Qwen2.5 32B Base	Qwen2.5 72B Base	DeepSeek V2 Base	DeepSeek V3 Base	dots.llm1 .base
	Architecture	-	Dense	Dense	MoE	MoE	MoE
	# Activated Params	-	32B	72B	21B	37B	14B
	# Total Params	-	32B	72B	236B	671B	142B
Chinese	C-Eval (EM)	5-shot	87.4	89.3	80.6	89.1	92.8
	CMMLU (EM)	5-shot	88.5	89.7	82.9	88.1	90.4
	CLUEWSC (EM)	5-shot	93.2	93.2	92.2	92.6	92.7
	C3 (EM)	5-shot	97.2	97.1	96.7	97.4	97.6
	Xiezhi (EM)	3-shot	81.2	82.3	77.4	80.4	82.9
	Average		89.5	90.3	86.0	89.5	91.3
English	TriviaQA (EM)	5-shot	78.5	85.2	89.2	91.4	88.2
	BBH (EM)	3-shot	82.2	83.7	76.5	83.5	80.8
	MMLU (EM)	5-shot	83.5	85.4	78.0	86.8	83.2
	MMLU-Pro (EM)	5-shot	61.8	64.7	52.3	61.4	61.9
	SuperGPQA (EM)	5-shot	30.5	32.1	29.0	38.6	33.3
	DROP (F1)	3-shot	82.9	84.9	78.9	89.1	83.4
	ARC-Challenge (EM)	25-shot	93.1	95.2	92.1	96.2	93.8
	AGIEval (EM)	3-shot	76.1	76.9	68.34	77.1	79.9
	HellaSwag (EM)	10-shot	93.3	94.2	88.1	89.7	88.2
	PIQA (EM)	5-shot	92.8	94.5	90.9	94.0	91.0
	NaturalQuestions (EM)	3-shot	35.3	42.7	46.8	50.7	48.7
	Average		73.2	76.3	71.8	78.0	75.7
MATH	GSM8K (EM)	8-shot	88.9	89.9	80.0	90.1	86.7
	MATH (EM)	4-shot	71.7	76.2	42.3	72.8	68.7
	CMath (EM)	3-shot	79.2	65.7	72.2	83.3	79.5
	Average		79.9	77.3	64.8	82.1	78.3
Code	HumanEval (Pass@1)	0-shot	50.6	56.7	43.3	62.8	64.0
	MBPP (Pass@1)	3-shot	77.0	77.8	75.1	76.3	76.7
	MCEval (Pass@1)	0-shot	47.9	47.5	43.5	50.1	46.6
	BigCodeBench (Pass@1)	0-shot	51.9	54.0	41.8	60.7	50.9
	Average		56.9	59.0	50.9	62.5	59.6

For English closed-book question answering, we utilize TriviaQA (Joshi et al., 2017) and Natural Questions (Kwiatkowski et al., 2019). For Chinese language understanding, we assess reference disambiguation using CLUEWSC (Xu et al., 2020) and reading comprehension using C3 (Sun et al., 2019).

Knowledge   To measure domain knowledge in English, we leverage the MMLU (Hendrycks et al., 2021a), MMLU-Pro (Wang et al., 2024b) and SuperGPQA (Team et al., 2025) benchmarks as well as for creative problem-solving and AGIEval (Zhong et al., 2023) for standardized exam performance. For the assessment of Chinese knowledge, we employ C-Eval (Huang et al., 2023), CMMLU (Li et al., 2023), and Xiezhi (Gu et al., 2024).

Mathematics   To evaluate mathematical reasoning capabilities, we utilize CMath (Wei et al., 2023) and GSM8K (Cobbe et al., 2021) for foundational problem-solving tasks, as well as MATH (Hendrycks et al., 2021b) for assessing advanced mathematical problem-solving proficiency.

Code   Code generation skills are evaluated with HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021) and MCEval (Chai et al., 2024). Additionally, the ability to handle diverse function calls and complex instructions is measured through the BigCodeBench benchmark (Zhuo et al., 2025).

4.4.2   Results

As shown in Table 2, we compare `dots.llm1` with other leading open-source base models under identical conditions. The results demonstrate that `dots.llm1`, with only 14B activated parameters, achieves superior performance compared to DeepSeek-V2, and delivers results comparable to Qwen2.5 72B. This makes `dots.llm1` one of the most economical and powerful mixture-of-experts models available to date.

For a detailed breakdown across different categories, we use Qwen2.5 72B as the baseline for comparison.
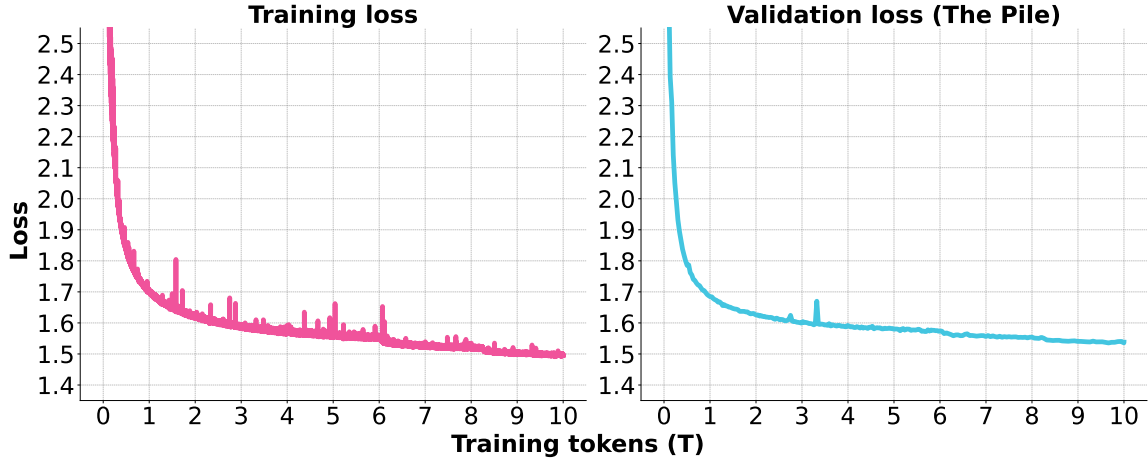
Figure 3: The loss curve highlights the consistent stability of the training process. At 6 trillion training tokens, we adjusted our batch size from 64 million to 96 million. At 8.3 trillion, we further increased it to 128 million.

Table 3: Long Context Performance on the RULER benchmark.

Model	4K	8K	16K	32K
Qwen2.5 72B	96.5	94.3	93.1	92.7
dots.llm1	94.7	94.9	92.6	87.7

dots.llm1 demonstrates comparable performance to Qwen2.5 72B across most domains. (1) On language understanding tasks, dots.llm1 achieves superior performance on Chinese language understanding benchmarks, which can be attributed to our data processing pipeline. (2) In knowledge tasks, while dots.llm1 shows slightly lower scores on English knowledge benchmarks, its performance on Chinese knowledge tasks remains robust. (3) In the code and mathematics domains, dots.llm1 achieves higher scores on HumanEval and CMath. Interestingly, for mathematics, we observe that dots.llm1 achieves better performance in the zero-shot setting compared to the few-shot setting, with an improvement of over 4 points. We leave further investigation of this phenomenon to future work.

4.5 Analysis

Stable Training   As depicted in Figure 3, the loss curve highlights the remarkable stability of the training process. Throughout the training period, there are no incidents of irrecoverable loss spikes or the need for rollback operations.

High Quality Web Data   To assess the quality of our web data, we conduct validation experiments comparing it to the TxT360 dataset (Tang et al., 2024), which represents the current state-of-the-art (SOTA) in open-source web data. The experiments are performed using a 1.5 billion parameter dense model, with the same architecture as Qwen2.5 1.5B. We randomly sample 350 billion tokens for training to obtain reliable experimental results. As illustrated in Figure 4, our results surpass those of TxT360, demonstrating the high quality of our web data through our processing pipeline.

Effective Long Context Extension   We use RULER (Hsieh et al., 2024) on the base models to evaluate long-context capabilities. As shown in Table 3, dots.llm1 delivers competitive performance at both 8K and 16K context lengths.

Economical Training Cost   As highlighted in Table 4, dots.llm1 showcases significant efficiency in terms of training costs. The comparison reveals that our approach substantially reduces the GPU hours required per one trillion tokens. Qwen2.5 72B model (in our optimized framework) requires 340K GPU hours, while dots.llm1 reduces this to just 130K GPU hours, leading to substantial savings in computational resources and expenses. Furthermore, if we consider the entire pretraining process, dots.llm1 requires only 1.46 million GPU hours, whereas Qwen2.5 72B consumes 6.12 million GPU hours, representing a reduction of $4\times$ in total computational resources. This substantial gap demonstrates the cost-effectiveness and scalability of dots.llm1, making it a more economical choice for large-scale pretraining.
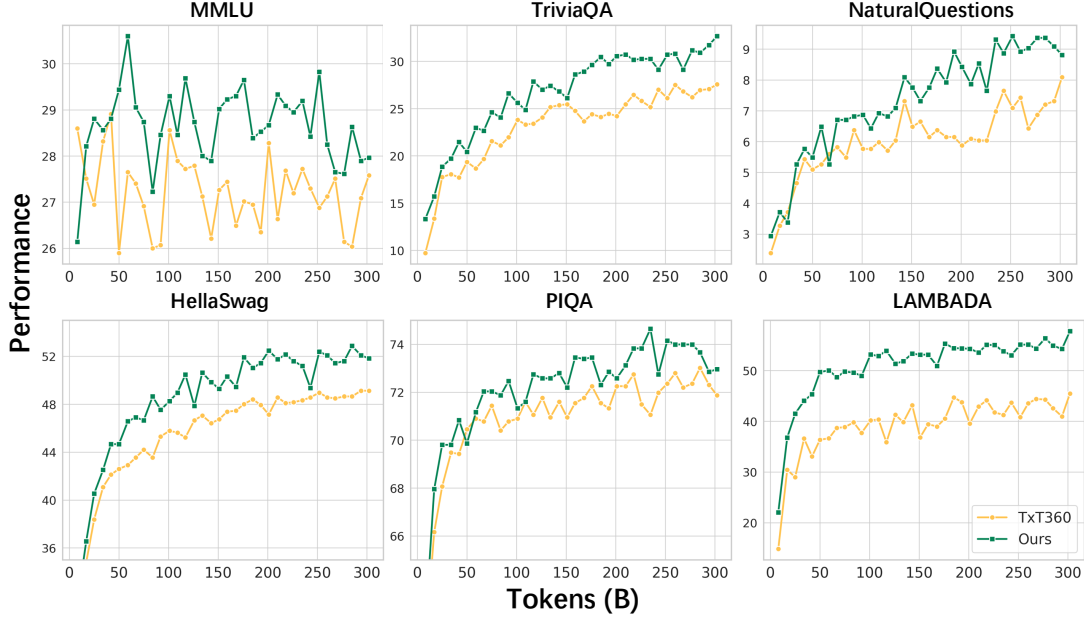
7

Figure 4: Comparison of performance curves between the TxT360 dataset and our web data on the MMLU, TriviaQA, NaturalQuestions, HellaSwag, PIQA, and LAMBADA (Paperno et al., 2016) benchmarks. The results demonstrate that training with our web data consistently yields superior performance.

Table 4: Comparison of GPU hours across different models, dots.llm1 representing a reduction of $4\times$ in total computational resources.

Model	GPU Hours per One Trillion Tokens	Pretraining Tokens	Total GPU Hours
Qwen2.5 72B	340K ($\times 1.0$)	18.0T	6,120K ($\times 1.0$)
dots.llm1	130K ($\times 0.38$)	11.2T	1,456K ($\times 0.24$)

5 Post-training

5.1 Supervised Fine-tuning

Data Mixture Based on open-source data and internally annotated data, we collect approximately 400K instruction tuning instances, focusing on several key areas: multi-lingual (primarily Chinese and English) multi-turn dialogues, knowledge understanding and question answering, complex instruction following, and reasoning tasks involving mathematics and coding.

For multi-turn dialogue capabilities, we integrate mainstream open-source dialogue datasets with our extensive collection of high-quality internal Chinese instruction sets. For a small subset of responses in the open-source data that are of insufficient quality, we refine them using powerful teacher models (e.g., DeepSeek-V3 0324 (DeepSeek-AI et al., 2024)). To enhance knowledge understanding QA abilities, we specifically incorporate datasets focused on factual knowledge and reading comprehension. For complex instruction following, we meticulously design instruction sets with conditional constraints and filtered out responses that failed to adhere to these conditions. To develop reasoning capabilities, we introduce verifiable mathematics and coding datasets, equipped with corresponding verifiers to filter and extract high-quality supervision signals.

Fine-tuning Configurations The fine-tuning process of dots.llm1.inst consists of two phases. In the first phase, we perform upsampling and multi-session concatenation on our 400K instruction tuning instances, then fine-tune dots.llm1.inst for 2 epochs. In the second phase, we further enhance the model's capabilities in specific domains (such as mathematics and coding) through rejection sampling fine-tuning (RFT), incorporating a verifier system to improve performance in these specialized areas. During each training phase, we employ a cosine learning rate scheduler with a learning rate of 5e-6 that gradually decayed to a minimum of 1e-6.

5.2 Evaluations

Benchmarks We evaluate dots.llm1.inst on a series of post-training benchmarks and compared it with state-of-the-art models (DeepSeek-AI, 2024b; DeepSeek-AI et al., 2024). Our evaluation framework covers five main categories: English general capabilities, Chinese general capabilities, alignment, and two specific subdomains—mathematics and coding.

For English general capabilities, we primarily use representative QA datasets, such as general question-answering benchmarks like MMLU (Hendrycks et al., 2021a), MMLU-Redux (Gema et al., 2024), and MMLU-Pro (Wang et al., 2024b), expert-level question-answering benchmarks like GPQA (Rein et al., 2023), as well as reading comprehension QA datasets such as DROP (Dua et al., 2019). Additionally, SimpleQA (OpenAI, 2024) evaluates model's ability to answer fact-seeking questions correctly.

For Chinese general capabilities, we select CLUEWSC (Xu et al., 2020) for coreference resolution evaluation, C-Eval (Huang et al., 2023) for comprehensive Chinese language understanding capabilities, and C-SimpleQA (He et al., 2024) for factual question answering in Chinese.

To measure the instruction-following and human preference alignment of the dots.llm1.inst model, we select rule-based verifiable benchmarks (IFEval (Zhou et al., 2023)) and competitive arena-style benchmarks (AlpacaEval2 (Dubois et al., 2024), ArenaHard (Li et al., 2024)).

For mathematics, we evaluate on competition-level benchmarks AIME24 (MAA, 2024) and CNMO24 (Liu et al., 2024b), and standard-difficulty benchmarks MATH (Hendrycks et al., 2021b), GSM8K (Cobbe et al., 2021), and MATH500 (Lightman et al., 2023).

For coding capabilities, we conduct tests on the coding competition LiveCodeBench (Jain et al., 2024), as well as HumanEval (Chen et al., 2021) and MBPP+ (Liu et al., 2024a).

Evaluation Configurations For general question-answering benchmarks (including MMLU-Pro, DROP, GPQA, SimpleQA, and Chinese SimpleQA), we utilize evaluation prompts provided by the simple-evals[1] framework, while for MMLU-Redux, we apply the zero-shot prompt template from the Zero-Eval[2] repository. For other general and alignment datasets, we follow the evaluation methodologies designed by the original dataset creators.

In mathematics evaluations, we adhere to standard practices for mathematical datasets, using "Please reason step by step, and put your final answer within \boxed{}" for answer extraction. For competitive mathematics benchmarks (e.g., AIME24), we employ temperature sampling and multiple-run averaging to enhance evaluation stability. In our code evaluations for LiveCodeBench, we select data spanning from May 2024 to January 2025. For evaluating MBPP+, we utilized the v0.2.0 release of EvalPlus (Liu et al., 2023), which provides reliable assessment for code generation tasks.

5.3 Performance Analysis

English Performance Our evaluation results indicate that dots.llm1.inst demonstrates stable and comprehensive performance across multiple general English benchmarks. On question-answering tasks such as MMLU, MMLU-Redux, DROP, and GPQA, dots.llm1.inst delivers competitive results compared to the Qwen2.5 / Qwen3 series models, demonstrating solid knowledge foundations and reasoning abilities.

Code Performance The model exhibits competitive coding capabilities, though this aspect is not quite as exceptional as its performance in other areas. Compared to the Qwen2.5 series, its coding ability is on par; however, in comparison with more cutting-edge models like Qwen3 and DeepSeek-V3, there remains room for further improvement.

Math Performance The dots.llm1.inst showcases outstanding mathematical reasoning skills. The model achieves a score of 33.1 on AIME24, underscoring its advanced problem-solving abilities in complex mathematics. On MATH500, it attains a score of 84.8, outperforming the Qwen2.5 series and approaching state-of-the-art results. It also achieved a high score of 40.6 on CNMO24.

Chinese Performance The dots.llm1.inst demonstrates significant advantages in Chinese language tasks. It achieves a score of 92.6 on CLUEWSC, matching industry-leading performance in Chinese semantic understanding. On C-Eval, it achieves 92.2, surpassing all models including DeepSeek-V3

Table 5: Comparison between dots.llm1.inst and other representative instruction-tuned models. All models were evaluated within our internal framework under identical conditions. Regarding the Qwen3 series, our evaluations were conducted without the thinking mode enabled.

	Benchmark (Metric)	Qwen-2.5 32B Inst	Qwen-2.5 72B Inst	Qwen-3 32B	Qwen-3 235B-A22B	DeepSeek V2 Chat	DeepSeek V3	gpt4o 0806	dots.llm1 .inst
	Architecture	Dense	Dense	Dense	MoE	MoE	MoE	-	MoE
	# Activated Params	32B	72B	32B	22B	21B	37B	-	14B
	# Total Params	32B	72B	32B	235B	236B	671B	-	142B
English	MMLU (EM)	83.4	84.4	82.9	86.4	78.6	87.9	86.7	82.1
	MMLU-Redux (EM)	83.3	85.9	86.1	88.6	77.9	88.8	87.3	85.1
	MMLU-Pro (EM)	68.9	71.0	68.3	69.9	60.2	76.0	74.4	70.4
	DROP (F1)	77.8	76.9	88.6	84.8	73.3	91.8	87.6	87.0
	GPQA Diamond (Pass@1)	47.2	49.9	54.5	59.4	32.6	56.1	50.1	52.6
	SimpleQA (Correct)	6.3	9.6	6.7	12.1	12.1	24.6	38.8	9.3
	Average	61.2	62.9	64.5	66.9	55.8	70.9	70.8	64.4
Code	HumanEval (Pass@1)	87.8	84.8	90.2	90.2	79.3	91.5	92.1	88.4
	MBPP+ (Pass@1)	74.6	74.6	74.9	75.9	67.2	75.7	75.9	74.5
	LiveCodeBench (Pass@1)	31.2	32.7	36.7	43.5	22.0	42.7	38.1	32.0
	Average	64.5	64.0	67.3	69.9	56.2	70.0	68.7	65.0
Math	MATH (EM)	82.9	83.6	86.2	88.6	57.1	89.7	79.4	85.0
	AIME24 (Pass@1)	16.5	18.8	27.1	37.5	2.9	34.0	13.3	33.1
	MATH500 (Pass@1)	81.9	83.1	84.8	88.1	56.2	88.9	78.4	84.8
	CNMO24 (Pass@1)	21.1	21.9	26.0	33.2	4.3	33.9	15.6	40.6
	Average	50.6	51.9	56.0	61.9	30.1	61.6	46.7	60.9
Chinese	CLUEWSC (EM)	91.7	93.1	91.6	91.4	93.0	93.5	90.3	92.6
	C-Eval (EM)	87.4	88.5	82.8	84.4	78.4	86.3	77.8	92.2
	C-SimpleQA (Correct)	42.3	51.4	46.6	62.0	56.5	68.9	61.4	56.7
	Average	73.8	77.7	73.7	79.3	76.0	82.9	76.5	80.5
Alignment	IFEval (Prompt Strict)	78.9	84.8	84.1	84.1	58.4	86.1	85.2	82.1
	AlpacaEval2 (GPT-4o Judge)	50.0	51.3	58.1	66.8	40.4	66.5	53.6	64.4
	ArenaHard (GPT-4o Judge)	76.7	85.6	91.5	95.7	41.8	92.1	85.1	87.1
	Average	68.5	73.9	77.9	82.2	46.9	81.6	74.6	77.9

(86.3), thus demonstrating comprehensive mastery of Chinese knowledge. For C-SimpleQA, it scores 56.7, substantially outperforming Qwen-2.5 72B Instruct, though still behind DeepSeek-V3 (68.9).

Alignment Performance   With respect to instruction following and alignment with human preferences, dots.llm1.inst demonstrates competitive performance on benchmarks such as IFEval, AlpacaEval2, and ArenaHard. These results indicate that the model can accurately interpret and execute complex instructions while maintaining consistency with human intentions and values.

Overall   Based on the evaluation results, dots.llm1.inst demonstrates exceptional performance across Chinese and English general tasks, mathematical reasoning, code generation, and alignment benchmarks while activating only 14B parameters. The model exhibits strong competitive advantages compared to Qwen2.5-32B-Instruct and Qwen2.5-72B-Instruct, and achieves comparable or superior performance relative to Qwen3-32B in bilingual tasks, mathematical reasoning, and alignment capabilities, establishing itself as a highly efficient open-source solution.

6   MoE Analysis

We examine the expert load of the dots.llm1 model on the Pile test set. For a specific domain D, with N_D tokens processed from this domain by the MoE, the expert load of an expert E_i is defined as:

$$\text{Expert Load}(E_i, D) = \frac{N_{E_i,D}}{N_D}, \quad \in [0, 1] \tag{1}$$

where $N_{E_i,D}$ represents the number of tokens from domain D that are routed to expert E_i. The expert load indicates how specialized expert E_i is for domain D. A value of 1 means that every token from the domain is routed to the expert, while a value of 0 implies the expert is never used for that domain.

dots.llm1 shows stronger expert specialization across all layers. In Figure 5, we observe some experts that are activated significantly more than random selection for specific domains, such as DM Mathematics. In contrast, compared to DM Mathematics, the expert load in Wikipedia appears more balanced. This could be because Wikipedia contains a wide range of knowledge and acts like a knowledge graph
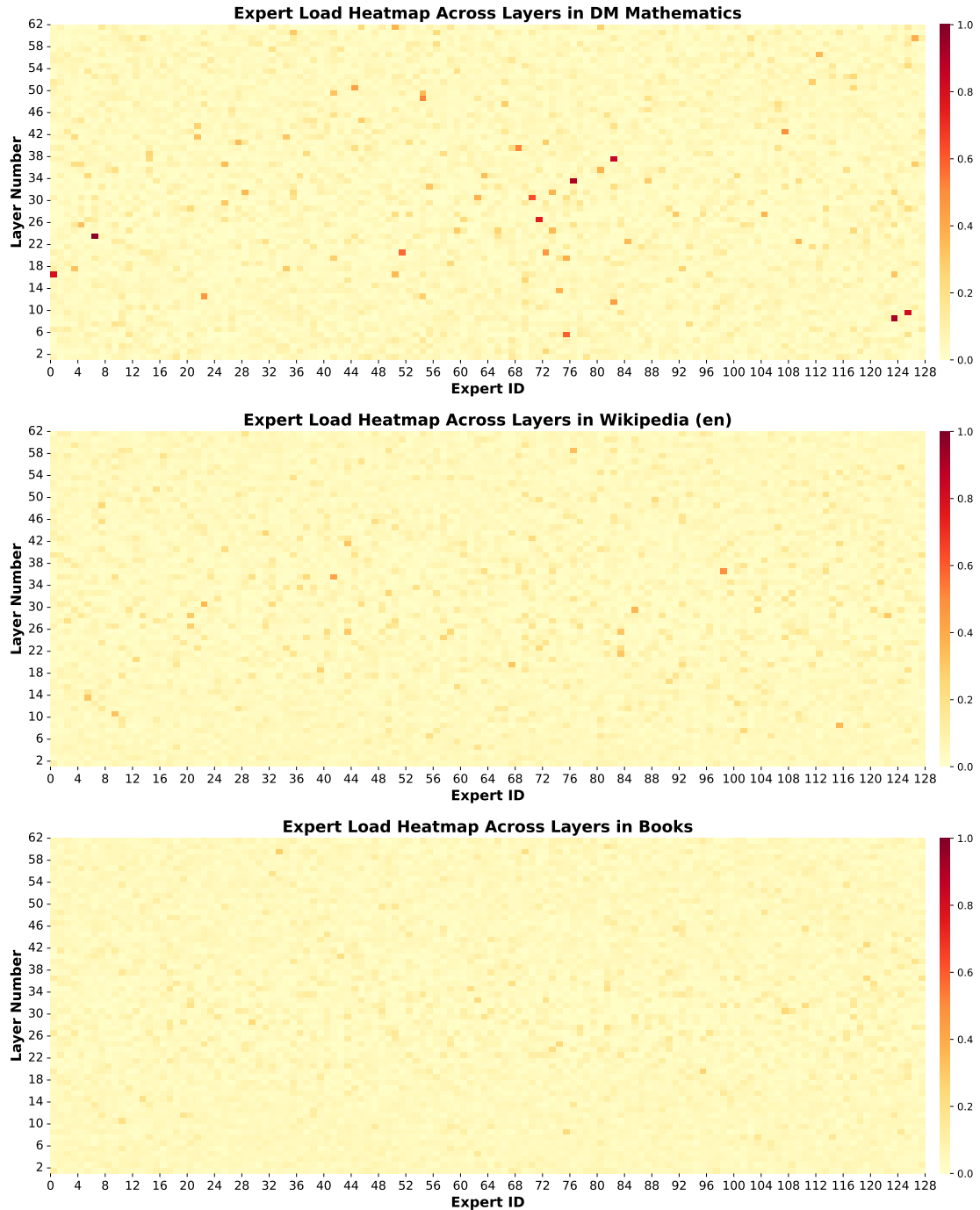
Figure 5: Expert load heatmap across layers on the Pile test dataset.

encompassing various entities. For the Books domain, which contains various types of data, the expert loads appear more balanced. This highlights that the load balancing function works as intended, enabling the model to effectively utilize all experts for handling generic data.

7 Conclusion and Future Work

We introduce `dots.llm1`, a cost-efficient mixture-of-experts model that achieves state-of-the-art performance for its size. By activating only a subset of parameters per token, `dots.llm1` significantly reduces training costs while delivering results comparable with much larger models. Our advanced data processing pipeline produces high-quality training data, and the open-sourcing of intermediate checkpoints provides valuable insights into model learning dynamics. `dots.llm1` demonstrates that efficient design and high-quality data can continually expand the capability boundaries of large language models.

As part of our future work, we aim to train a more powerful model. To achieve an optimal balance between training and inference efficiency, we plan to integrate more efficient architectural designs such as Grouped Query Attention (GQA) (Ainslie et al., 2023), Multi-Head Latent Attention (MLA) (DeepSeek-AI, 2024b), and Linear Attention (Yang et al., 2023; Katharopoulos et al., 2020). Additionally, we intend to explore the use of more sparse mixture-of-experts (MoE) layers to improve the computational efficiency. Furthermore, since data serves as the foundation of pretraining, we will deepen our understanding of what constitutes optimal training data and explore methods to achieve more human-like learning efficiency, maximizing knowledge acquisition from each training example.

References

AI@Meta. Llama 3 model card, 2024a. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

AI@Meta. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation, 2024b. URL https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

Anthropic. Claude 3.7 sonnet system card, 2025. URL https://assets.anthropic.com/m/785e231869ea8b3b/original/claude-3-7-sonnet-system-card.pdf.

Jacob Austin, Augustus Odena, Maxwell I. Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie J. Cai, Michael Terry, Quoc V. Le, and Charles Sutton. Program synthesis with large language models. *CoRR*, abs/2108.07732, 2021.

Adrien Barbaresi. Trafilatura: A Web Scraping Library and Command-Line Tool for Text Discovery and Extraction. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pp. 122–131. Association for Computational Linguistics, 2021. URL https://aclanthology.org/2021.acl-demo.15.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 7432–7439. AAAI Press, 2020. doi: 10.1609/aaai.v34i05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

Andrei Z Broder. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, pp. 21–29. IEEE, 1997.

Linzheng Chai, Shukai Liu, Jian Yang, Yuwei Yin, Ke Jin, Jiaheng Liu, Tao Sun, Ge Zhang, Changyu Ren, Hongcheng Guo, Zekun Wang, Boyang Wang, Xianjie Wu, Bing Wang, Tongliang Li, Liqun Yang, Sufeng Duan, and Zhoujun Li. Mceval: Massively multilingual code evaluation, 2024. URL https://arxiv.org/abs/2406.07436.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harrison 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, Joshua 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. *CoRR*, abs/2107.03374, 2021.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *CoRR*, abs/1803.05457, 2018.

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. *CoRR*, abs/2110.14168, 2021.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *CoRR*, abs/2401.06066, 2024. URL https://doi.org/10.48550/arXiv.2401.06066.

DeepSeek-AI. Deepseek LLM: scaling open-source language models with longtermism. *CoRR*, abs/2401.02954, 2024a. URL https://doi.org/10.48550/arXiv.2401.02954.

DeepSeek-AI. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *CoRR*, abs/2405.04434, 2024b. URL https://doi.org/10.48550/arXiv.2405.04434.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2024. URL https://arxiv.org/abs/2412.19437.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning*, pp. 7480–7512. PMLR, 2023.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 2368–2378. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1246. URL https://doi.org/10.18653/v1/n19-1246.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, 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, 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, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, 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, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The Llama 3 herd of models. *CoRR*, abs/2407.21783, 2024.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 23:120:1–120:39, 2022.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile van Krieken, and Pasquale Minervini. Are we done with mmlu? *CoRR*, abs/2406.04127, 2024. URL https://doi.org/10.48550/arXiv.2406.04127.

Zhouhong Gu, Xiaoxuan Zhu, Haoning Ye, Lin Zhang, Jianchen Wang, Yixin Zhu, Sihang Jiang, Zhuozhi Xiong, Zihan Li, Weijie Wu, Qianyu He, Rui Xu, Wenhao Huang, Jingping Liu, Zili Wang, Shusen Wang, Weiguo Zheng, Hongwei Feng, and Yanghua Xiao. Xiezhi: An ever-updating benchmark for holistic domain knowledge evaluation, 2024. URL https://arxiv.org/abs/2306.05783.

Yancheng He, Shilong Li, Jiaheng Liu, Yingshui Tan, Weixun Wang, Hui Huang, Xingyuan Bu, Hangyu Guo, Chengwei Hu, Boren Zheng, et al. Chinese simpleqa: A chinese factuality evaluation for large language models. *arXiv preprint arXiv:2411.07140*, 2024.

Babak Hejazi. Introducing grouped gemm apis in cublas and more performance updates, June 2024. URL https://developer.nvidia.com/blog/introducing-grouped-gemm-apis-in-cublas-and-more-performance-updates/.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*. OpenReview.net, 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. In *NeurIPS Datasets and Benchmarks*, 2021b.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? *CoRR*, abs/2404.06654, 2024.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zhen Leng Thai, Kai Zhang, 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. *CoRR*, abs/2404.06395, 2024.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*, 2023.

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

Ahad Jawaid, Shreeram Suresh Chandra, Junchen Lu, and Berrak Sisman. Style mixture of experts for expressive text-to-speech synthesis. *arXiv preprint arXiv:2406.03637*, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet,

Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts. *CoRR*, abs/2401.04088, 2024.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://doi.org/10.1162/tacl_a_00276.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive data sets*. Cambridge university press, 2020.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. CMMLU: Measuring massive multitask language understanding in Chinese. *CoRR*, abs/2306.09212, 2023.

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

Xiuhong Li, Yun Liang, Shengen Yan, Liancheng Jia, and Yinghan Li. A coordinated tiling and batching framework for efficient GEMM on gpus. In Jeffrey K. Hollingsworth and Idit Keidar (eds.), *Proceedings of the 24th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP 2019, Washington, DC, USA, February 16-20, 2019*, pp. 229–241. ACM, 2019. doi: 10.1145/3293883.3295734. URL https://doi.org/10.1145/3293883.3295734.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

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. *Advances in Neural Information Processing Systems*, 36:21558–21572, 2023.

Jiawei Liu, Songrun Xie, Junhao Wang, Yuxiang Wei, Yifeng Ding, and Lingming Zhang. Evaluating language models for efficient code generation. *arXiv preprint arXiv:2408.06450*, 2024a.

Junnan Liu, Hongwei Liu, Linchen Xiao, Ziyi Wang, Kuikun Liu, Songyang Gao, Wenwei Zhang, Songyang Zhang, and Kai Chen. Are your llms capable of stable reasoning? *arXiv preprint arXiv:2412.13147*, 2024b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *CoRR*, abs/1711.05101, 2017.

MAA. American invitational mathematics examination - aime. In *American Invitational Mathematics Examination - AIME 2024*, February 2024. URL https://maa.org/math-competitions/american-invitational-mathematics-examination-aime.

James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, volume 5, pp. 281–298. University of California press, 1967.

Mistral. Mistral small 3, 2025. URL https://mistral.ai/news/mistral-small-3.

NVIDIA. pytorch - transformer engine 2.1.0 documentation, August 2024a. URL https://docs.nvidia.com/deeplearning/transformer-engine/user-guide/api/pytorch.html#transformer_engine.pytorch.GroupedLinear.

NVIDIA. MoE A2A Interleaved 1F1B based Computation and Communication Overlap, 2024b. URL https://developer.nvidia.com/zh-cn/blog/1f1b-moe-a2a-computing-overlap/. Accessed: 2025-04-23.

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

OpenAI. Introducing SimpleQA, 2024. URL https://openai.com/index/introducing-simpleqa/.

OpenAI. Introducing gpt-4.5, 2025a. URL https://openai.com/index/introducing-gpt-4-5/.

OpenAI. Introducing openai o3 and o4-mini, 2025b. URL https://openai.com/index/introducing-o3-and-o4-mini/.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016. doi: 10.18653/v1/p16-1144. URL https://doi.org/10.18653/v1/p16-1144.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *CoRR*, abs/2306.01116, 2023.

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*, 2024. URL https://openreview.net/forum?id=n6SCkn2QaG.

Qwen. Qwen2.5: A party of foundation models, 2024a. URL https://qwenlm.github.io/blog/qwen2.5.

Qwen. Qwen1.5-moe: Matching 7b model performance with 1/3 activated parameters", February 2024b. URL https://qwenlm.github.io/blog/qwen-moe/.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446, 2021.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. *CoRR*, abs/2311.12022, 2023.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. *arXiv preprint arXiv:2409.16040*, 2024.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. Investigating prior knowledge for challenging chinese machine reading comprehension, 2019.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *ACL (Findings)*, pp. 13003–13051. Association for Computational Linguistics, 2023.

Liping Tang, Nikhil Ranjan, Omkar Pangarkar, Xuezhi Liang, Zhen Wang, Li An, Bhaskar Rao, Linghao Jin, Huijuan Wang, Zhoujun Cheng, Suqi Sun, Cun Mu, Victor Miller, Xuezhe Ma, Yue Peng, Zhengzhong Liu, and Eric P. Xing. Txt360: A top-quality llm pre-training dataset requires the perfect blend, 2024.

P Team, Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, King Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, Kaixin Deng, Shawn Gavin, Shian Jia, Sichao Jiang, Yiyan Liao, Rui Li, Qinrui Li, Sirun Li, Yizhi Li, Yunwen Li, David Ma, Yuansheng Ni, Haoran Que, Qiyao Wang, Zhoufutu Wen, Siwei Wu, Tyshawn Hsing, Ming Xu, Zhenzhu Yang, Zekun Moore Wang, Junting Zhou, Yuelin Bai, Xingyuan Bu, Chenglin Cai, Liang Chen, Yifan Chen, Chengtuo Cheng, Tianhao Cheng, Keyi Ding, Siming Huang, Yun Huang, Yaoru Li, Yizhe Li, Zhaoqun Li, Tianhao Liang, Chengdong Lin, Hongquan Lin, Yinghao Ma, Tianyang Pang, Zhongyuan Peng, Zifan Peng, Qige Qi, Shi Qiu, Xingwei Qu, Shanghaoran Quan, Yizhou Tan, Zili Wang, Chenqing Wang, Hao Wang, Yiya Wang, Yubo Wang, Jiajun Xu, Kexin Yang, Ruibin Yuan, Yuanhao Yue, Tianyang Zhan, Chun Zhang, Jinyang Zhang, Xiyue Zhang, Xingjian Zhang, Yue Zhang, Yongchi Zhao, Xiangyu Zheng, Chenghua Zhong, Yang Gao, Zhoujun Li, Dayiheng Liu, Qian Liu, Tianyu Liu, Shiwen Ni, Junran Peng, Yujia Qin, Wenbo Su, Guoyin Wang, Shi Wang, Jian Yang, Min Yang, Meng Cao, Xiang Yue, Zhaoxiang Zhang, Wangchunshu Zhou, Jiaheng Liu, Qunshu Lin, Wenhao Huang, and Ge Zhang. Supergpqa: Scaling llm evaluation across 285 graduate disciplines, 2025. URL https://arxiv.org/abs/2502.14739.

Junfeng Tian, Rui Wang, Cong Li, Yudong Zhou, Jun Liu, and Jun Wang. Nyonic technical report. *arXiv preprint arXiv:2404.15702*, 2024a.

Junfeng Tian, Da Zheng, Yang Cheng, Rui Wang, Colin Zhang, and Debing Zhang. Untie the knots: An efficient data augmentation strategy for long-context pre-training in language models, 2024b. URL https://arxiv.org/abs/2409.04774.

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

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *CoRR*, abs/2408.15664, 2024a. URL https://doi.org/10.48550/arXiv.2408.15664.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *CoRR*, abs/2406.01574, 2024b. URL https://doi.org/10.48550/arXiv.2406.01574.

Tianwen Wei, Jian Luan, Wei Liu, Shuang Dong, and Bin Wang. Cmath: Can your language model pass chinese elementary school math test?, 2023.

Mitchell Wortsman, Peter J Liu, Lechao Xiao, Katie Everett, Alex Alemi, Ben Adlam, John D Co-Reyes, Izzeddin Gur, Abhishek Kumar, Roman Novak, et al. Small-scale proxies for large-scale transformer training instabilities. *arXiv preprint arXiv:2309.14322*, 2023.

xAI. Grok 3 beta — the age of reasoning agents, 2025. URL https://x.ai/news/grok-3.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, Yin Tian, Qianqian Dong, Weitang Liu, Bo Shi, Yiming Cui, Junyi Li, Jun Zeng, Rongzhao Wang, Weijian Xie, Yanting Li, Yina Patterson, Zuoyu Tian, Yiwen Zhang, He Zhou, Shaoweihua Liu, Zhe Zhao, Qipeng Zhao, Cong Yue, Xinrui Zhang, Zhengliang Yang, Kyle Richardson, and Zhenzhong Lan. CLUE: A chinese language understanding evaluation benchmark. In Donia Scott, Núria Bel, and Chengqing Zong (eds.), *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pp. 4762–4772. International Committee on Computational Linguistics, 2020. doi: 10.18653/V1/2020.COLING-MAIN.419. URL https://doi.org/10.18653/v1/2020.coling-main.419.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*, 2023.

Zhiwei Yang, Lu Lu, and Ruimin Wang. A batched GEMM optimization framework for deep learning. *J. Supercomput.*, 78(11):13393–13408, 2022. doi: 10.1007/S11227-022-04336-3. URL https://doi.org/10.1007/s11227-022-04336-3.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David R. Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28-August 2, 2019, Volume 1: Long Papers*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1472. URL https://doi.org/10.18653/v1/p19-1472.

Yaqing Zhang, Yaobin Wang, Zhangbin Mo, Yong Zhou, Tao Sun, Guang Xu, Chaojun Xing, and Liang Yang. Accelerating small matrix multiplications by adaptive batching strategy on GPU. In *24th IEEE Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application, HPCC/DSS/SmartCity/DependSys 2022, Hainan, China, December 18-20, 2022*, pp. 882–887. IEEE, 2022. doi: 10.1109/HPCC-DSS-SMARTCITY-DEPENDSYS57074.2022.00143. URL https://doi.org/10.1109/HPCC-DSS-SmartCity-DependSys57074.2022.00143.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. AGIEval: A human-centric benchmark for evaluating foundation models. *CoRR*, abs/2304.06364, 2023. doi: 10.48550/arXiv.2304.06364. URL https://doi.org/10.48550/arXiv.2304.06364.

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

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, Simon Brunner, Chen Gong, Thong Hoang, Armel Randy Zebaze, Xiaoheng Hong, Wen-Ding Li, Jean Kaddour, Ming Xu, Zhihan Zhang, Prateek Yadav, Naman Jain, Alex Gu, Zhoujun Cheng, Jiawei Liu, Qian Liu, Zijian Wang, Binyuan Hui, Niklas Muennighoff, David Lo, Daniel Fried, Xiaoning Du, Harm de Vries, and Leandro Von Werra. Big-codebench: Benchmarking code generation with diverse function calls and complex instructions, 2025. URL https://arxiv.org/abs/2406.15877.

Appendix

A Authors

Bi Huo	Fu Guo	Shuo Liu	Yawen Liu
Bin Tu	Jian Yao	Su Guang	Yuqiu Ji
Cheng Qin	Jie Lou	Te Wo	Ze Wen
Da Zheng	Junfeng Tian	Weijun Zhang	Zhenhai Liu
Debing Zhang	Li Hu	Xiaoming Shi	Zichao Li
Dongjie Zhang	Ran Zhu	Xinxin Peng	Zilong Liao
En Li	Shengdong Chen	Xing Wu	

B Hyperparameters

We display the pretraining hyperparameter configuration in Table 6 comparing with other relevant models.

Table 6: Architectural and pretraining hyper-parameters of dots.llm1 compared with recent models. WSD = weight-stable decay (Hu et al., 2024).

	Qwen2.5 72B	DeepSeek-V2	DeepSeek-V3	dots.llm1
Architecture	Dense	MoE	MoE	MoE
Active parameters	72B	21B	37B	14B
Total parameters	72B	236B	671B	142B
Hidden size	8,192	5,120	7,168	4,096
Activation	SwiGLU	SwiGLU	SwiGLU	SwiGLU
FFN size	29,568	12,288	18,432	10,944
Vocabulary size	152,064	102,400	129,280	152,064
Attention type	GQA	MLA	MLA	MHA
Attention heads	64	128	128	32
KV heads	8	128	128	32
Layers	80	60	61	62
Attention biases	Yes	No	No	No
QK normalization	No	No	No	Yes
Positional encoding	RoPE	RoPE	RoPE	RoPE
MoE FFN size	–	1,536	2,048	1,408
Routed experts	–	160	256	128
Shared experts	–	2	1	2
Top-k experts	–	6	8	6
Init. std	–	0.006	0.006	0.006
LR schedule	–	WSD	WSD	WSD
Warmup steps	–	2,000	2,000	4,000
Peak LR	–	2.4E-4	2.2E-4	3.0E-4
Optimizer	–	AdamW	AdamW	AdamW
Weight decay	–	0.1	0.1	0.1
Beta1	–	0.9	0.9	0.9
Beta2	–	0.95	0.95	0.95
Pretraining tokens	18T	8.1T	14.8T	11.2T
Sequence length	4,096	4,096	4,096	8,192

C   Web Data Curation

C.1   Document Preparation Phase

URL Filtering   Similar to RefineWeb (Penedo et al., 2023), URL filtering focuses on filtering toxic documents based solely on the URL, rather than the document content. We maintain a blocklist of domains, implementing careful manual verification to ensure removal of pages associated with adult themes, gambling, and other toxic topics.

Text Extraction   Following RefineWeb (Penedo et al., 2023), we employ *trafilatura*(Barbaresi, 2021) to extract the main text from webpages. To mitigate the presence of irrelevant content, we implement a series of custom optimizations for *trafilatura*, including adjustments to HTML patterns, keyword filtering, and content length specifications.

Language Identification   Leveraging the fastText language identification model from CCNet (Joulin et al., 2016), we classify the language of each document efficiently. To ensure the quality and relevance of the dataset, we discard any documents with a language classification confidence score below a threshold of 0.65. This step is crucial for filtering out noisy or ambiguous data, particularly in multilingual datasets, and helps maintain a consistent and high-quality corpus.

Identity Removal   To ensure unique identification, we compute a 32-character hexadecimal string for each document using the MD5 digest algorithm. Duplicate documents are then removed randomly to retain only unique content for further processing. Prior to deduplication, we preprocess the text by removing all punctuation, normalizing characters to the NFD Unicode format[3], converting text to lowercase, and eliminating extra spaces between words. This step is essential for preserving dataset integrity and ensuring that only distinct documents are included in the final collection.

C.2   Rule-Based Processing Phase

Line-wise Inter-Document Deduplication   We introduce a line-level deduplication method specifically designed to target repetitive patterns commonly found in the head and tail sections of web documents, such as advertisements, navigation bars, and other non-informative content. This approach effectively reduces inter-document redundancy while preserving the essential and meaningful content, ensuring a cleaner and more focused dataset. First, we extract the first 5 lines and the last 5 lines of each document, splitting them into individual lines while discarding any lines that are empty or contain only whitespace or symbols. Next, we calculate the frequency of each line across the entire dataset. For any line that appears more than 200 times, we retain only its first 200 occurrences within the dataset and remove all additional instances from their respective documents. If a document does not contain any lines that need to be removed, it is preserved in its original form. Otherwise, the specified lines are removed, and the remaining parts are concatenated to form the final document. This method is implemented in an incremental manner and within distributed systems, making it more efficient and scalable for large-scale datasets.

Rule-Based Filtering   We develop a filtering system based on RefinedWeb (Penedo et al., 2023) and Gopher (Rae et al., 2021), combining precise heuristic rules and statistical features to systematically remove low-quality content. Our pipeline includes empty content removal, advertisement and registration prompt filtering, domain/URL/title-based meta filtering, wiki/code diff detection, structural anomaly and duplicate content elimination, and content quality filtering. Additionally, we perform manual review and tailored cleaning for major domains, focusing especially on the top 1,000 sites that comprise 60% of the data, to ensure consistently high data quality.

Fuzzy Deduplication   We employ MinHash (Broder, 1997) and Locality-Sensitive Hashing (LSH) (Leskovec et al., 2020) to perform approximate deduplication. The process involves the following steps: First, we apply the same text standardization as in the Identity Removal step. Next, we tokenize the text using the Jieba tokenizer [4] to handle Chinese text, followed by 5-gram processing. Using the resulting 5-gram tokens, we compute 2048 MinHash values for each text. These MinHash values are then divided into 128 bands, each containing 16 rows, as part of the LSH configuration. For any samples that collided within the same band, we retain only one instance. This approach enables us to efficiently deduplicate text pairs with a Jaccard similarity of 80%, achieving a high probability of 97.42%.

---

[3]https://en.wikipedia.org/wiki/Unicode_equivalence
[4]https://github.com/fxsjy/jieba

C.3 Model-Based Processing Phase

Web-Type Classification Model   Similar to how humans perceive and categorize information, we implement a web content classifier that distinguishes various types of web pages. The classifier utilizes a 1.5B model that categorizes content into text-rich detail pages (such as articles) and non-essential web pages, including tool pages (e.g., maps, public transit route queries), audio pages, video pages, forum sites, and adult websites. Through this process, we selectively preserve only high-quality detail pages for subsequent processing.

Web Clutter Removal Model   We implement a line-wise web clutter removal model to eliminate unnecessary web page elements such as borders, advertisements, navigational components, and repetitive content, focusing exclusively on main content. This model evaluates each individual line based on its informational value, relevance, and quality, assigning a score from 0 to 1. We fine-tune a 1.5B model and find that it significantly improves the overall data quality.

Quality Model   The quality model performs comprehensive multidimensional analysis to evaluate and score training samples (Qwen, 2024a; Penedo et al., 2024). We design a comprehensive annotation specification to evaluate text quality, including text fluency, coherence, information redundancy, and repetition patterns. In order to balance efficiency and model performance, we adopt a k-fold cross validation method with a 1.5B model and evaluate its discriminative ability using the area under the ROC curve (AUC). Using the quality score generated by this model, we set a threshold to only retain high-quality data from the corpus.

Semantic Deduplication   In line with Dubey et al. (2024), we implement a semantic deduplication strategy to eliminate documents with high semantic overlap. Initially, we employ `BGE-M3` (Chen et al., 2024) as an embedding model to create embeddings for each document. Subsequently, we apply the KMeans algorithm (MacQueen, 1967) to categorize these embeddings into clusters. This clustering step significantly reduces computational costs in subsequent pairwise similarity calculations. Following the clustering phase, we compute the pairwise cosine similarity between document embeddings within the same cluster. Duplicates are identified based on a predefined similarity threshold, which is set to 0.95 in our process. By doing so, we ensure the preservation of varied and distinctive content while discarding redundant semantic information.

Category Balancing   To curate a more balanced and information-rich training dataset, we first develop a fine-grained, 200-class content classifier to categorize the vast web data. This classifier enables us to identify and quantify the distribution of various content types within the corpus. Based on this categorization, we proportionally increase the representation of knowledge-based and factual content, such as encyclopedia entries, educational materials, and popular science articles. Meanwhile, we deliberately reduce the proportion of less informative or structurally formulaic content, including fiction (e.g., science fiction novels) and product descriptions. This balancing process ensures that the training dataset is both diverse and skewed towards sources that are rich in reliable information, thereby fostering the development of language models with stronger comprehension, reasoning, and knowledge retention abilities.