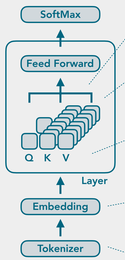
Model Anatomy



Grouped Query Attention

The 16 attention heads share 4 query which maintains the performance of full multi-head attention and saves memory during inference.

Intra-Document Masking

Attention masking ensures that tokens from different documents in the same training sequence don't attend to each other which helps with long context training.

NoRoPE

We use NoRoPE which selectively removing rotary position embeddings from every 4th layer improving long context performance.

No Weight Decay in Embeddings

We remove weight decay from embedding layers to improve training stability as embedding norms naturally stabilize at healthier values.

Multilingual Tokenizer

We use the Llama 3.2 tokenizer which covers all languages used for pretraining.

Training Configuration

Parameter count: 3.08B
Initialization: N(0, std=0.02)
Layers: 36
Rope theta: 50k

Sequence length: 4096
Batch size: 2.36M Tokens
Optimizer: AdamW (eps=1e-8, beta1=0.8, beta2=0.95)
Learning rate (peak): 2e-4
Gradient Clipping: 1.0
Weight decay: 0.1
Gradient accumulation: 1
Micro batch size: 3
Precision: bf16
Tensor parallel: 2
Data parallel: 192

Throughput: 14k tokens/sec/gpu
MFU: 29.43 %
Training duration: 24 days

SmolLM3: Blueprint

Small, multilingual, long-context reasoner

3B Params
dense model

Long Context
Up to 256k

Tools Use
code + json

Reasoning
dual think and no_think modes for reasoning

Tokenizer
Llama 3.2

Multilingual
6 languages: English, French, Spanish, German, Italian, Portuguese and support for several more

Open Source
Code, data, model are openly available

11T tokens
pretraining

Post-training
SFT + APO

Attention
GQA

Distributed Training

Training Cluster

48 nodes of 8 x H100 (80 GB) for **24 days** at the pretraining stage with a total of **220k GPUh**.

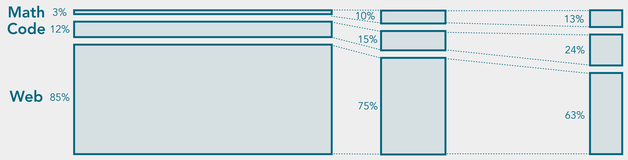
Distributed Layout

The model was split across 2 GPUs with Tensor Parallelism (TP=2) such that one node holds 4 model instances and then distributed across the 384 GPUs with Data Parallelism (DP=192).

Logging and Checkpointing

Training and evaluation metrics are logged to W&B while checkpoints are stored on S3 every 2k steps. Evaluation runs asynchronously independent of training

Pretraining Recipe



Phase I

Description: Base training
Duration: 8T tokens
Datasets: Base mix for pretraining
Web: FineWeb-Edu, DCLM, FineWeb2, FineWeb2-HQ
Code: The Stack v2 (16 langs), StarCoder2 PRs, Jupyter/Kaggle NBs, GH issues, StackExchange
Math: FineMath3+ | InfiWebMath3+

Phase II

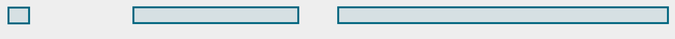
Description: High quality injection
Duration: 2T tokens
Datasets: Adding Stack-Edu, FineMath4+, InfiWebMath4+, MegaMath (incl. Qwen Q&A, Pro synthetic rewrites, and text code interleaved blocks)

Phase III

Description: LR Decay
Duration: 1.1T tokens
Datasets: Upsampling high quality code/math datasets and adding instruction/reasoning data such as OpenMathReasoning

LR Schedule

Long Context Training



Base: 4k

During pretraining a context length of 4k tokens was used. The long context training used the same data.
~8 pages of text

Step 1: 32k

The RoPE theta was increased to 1.5M and training continued for 50B tokens with 32k context size.
~64 pages of text

Step 2: 64k

The RoPE theta was further increased to 5M and training continued for 50B tokens with 64k context size.
~128 pages of text

YaRN: 128k

Using YaRN scaling on top of the 64k checkpoint allows to extend the context to 128k tokens.
~256 pages of text

Post-training Recipe

The post-training recipe starts of with the checkpoint after pretraining and long context adaptation. The optimizer state is not re-used from earlier training.

During mid-training the model is trained for **5 epochs** on a mix of OpenThoughts and Nemotron post-training data totalling **175B tokens** (35B unique).

The model is further trained on a mix of **25 high quality datasets** mixing reasoning/non-reasoning data for **4 epochs** and **10B tokens** (2.5B unique).

Finally, we linearly merge the model soup checkpoint with long context checkpoint using a 90/10 ratio to recover the long context capabilities.

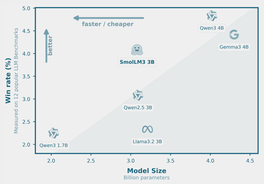
We found that model souping, where intermediate checkpoints from the APO stage are merged together further improves the downstream performance.

We apply Anchored Preference Optimization (APO), a variant of DPO, for 1 epoch with a thinking preference pair for every non-thinking preference pair.

Evaluation

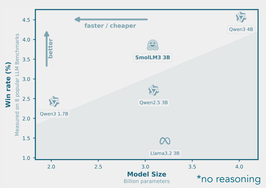
Base Model

	Hella Swag	ARC	MMLU Pro	MMLU BoolQ	MATH	Human Eval+	Global MMLU	Flores	Ruler 64k
Llama 3.2 3B	75.5	58.9	16.4	75.3	7.5	25.0	33.6	53.1	**72.9**
Qwen 2.5 3B	74.2	59.8	16.7	73.6	40.1	34.1	34.4	54.9	64.9
Qwen3 4B	74.4	62.1	**24.9**	74.3	**51.2**	**54.9**	**40.3**	55.8	60.3
SmolLM3 3B	**76.2**	**65.6**	19.6	**79.0**	46.1	30.5	37.2	**56.3**	67.9
	Knowledge & Reasoning				Math & Code		Multilingual fr/es/de/it/pt		Long Context



Instruct Model

		AIME 2025	GSM Plus	LCB v4	GPQA diam.	IFEval	MixEval Hard	MMLU Pro	Global MMLU
Reasoning	SmolLM3 3B	36.7	84.4	30.5	41.7	71.2	30.8	58.4	64.1
	Qwen3 1.7B	30.7	79.4	34.4	39.9	74.2	33.9	57.8	62.3
	Qwen3 4B	**58.8**	**88.2**	**52.9**	**55.3**	**85.4**	**38.0**	**70.2**	**73.3**
Non-Reasoning	SmolLM3 3B	9.3	72.8	15.2	35.7	**76.7**	26.9	45.0	53.5
	Qwen 2.5 3B	2.9	74.1	10.5	32.2	65.6	27.6	41.9	-
	Llama 3.2 3B	0.3	59.2	3.4	29.4	71.6	24.9	36.6	46.8
	Qwen3 1.7B	8.0	68.3	15.0	31.8	74.0	24.3	45.6	49.5
	Qwen3 4B	**17.1**	**82.1**	**24.9**	**44.4**	68.9	**31.6**	**60.9**	**65.1**



*no reasoning

Data Ablations and Synthetic Generation

From Scratch Ablations

Before starting the pretraining run we ran ablations to determine the appropriate data mix. Each ablation trained **from scratch for 100B tokens** to decide

- ratio for FineWeb-Edu / DCLM
- impact of datasets like wiki, pes2o, StackExchange
- fraction of multilingual data without hurting English
- percentage of code data used in pretraining

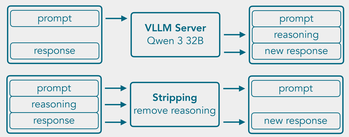
Annealing Ablations

Between pretraining phases at 8 and 10T we ran additional ablations to adjust the data mix. Each ablation was based on the **latest checkpoint and the LR was decayed for 50B tokens**. We investigated:

- mixing in high quality data for math and code
- impact of adding reasoning data at later stages

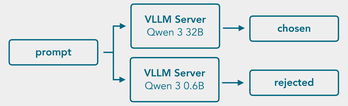
Synthetic SFT Data

For dual reasoning mode to work reliably it was necessary to generate synthetic data for datasets which had not reasoning traces and create a reasoning stripped version of reasoning datasets:



Synthetic Preference Data

For the APO step preference data is necessary which we generated using a strong (Qwen 3 32B) and a weak model (Qwen 3 0.6B):



Model Usage

Chat Template

transformers

```
from transformers import pipeline

model = "HuggingFaceTB/SmolLM3-3B"
pipe = pipeline("text-generation", model=model)

pipe("Is the earth flat?")
```

transformers-cli

```
transformers chat HuggingFaceTB/SmolLM3-3B
```

vllm

```
vllm serve "HuggingFaceTB/SmolLM3-3B" --model-impl transformers
```

llama.cpp

```
llama-server -hf HuggingFaceTB/SmolLM3-3B
```

mlx-lm

```
mlx_lm.chat --model "HuggingFaceTB/SmolLM3-3B"
```