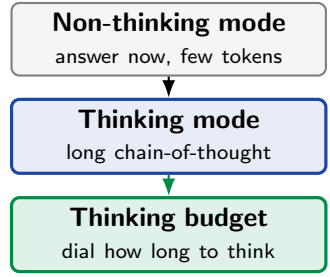
One model that can both think and answer fast

Until now the field split into **two kinds** of model: chat models that answer fast (GPT-4o), and dedicated *reasoning* models that think in long chains (o1, QwQ, DeepSeek-R1). You picked one *per request*.

Qwen3 folds both into *one* model. A flag in the prompt switches between **thinking** and **non-thinking** mode – and a **thinking budget** lets you dial *how many* tokens it may spend reasoning before it must answer.

More budget $\rightarrow$ higher accuracy, until it plateaus. A clean **compute-vs-quality knob**, exposed to the user.

Today: the family, the training, and how one model learns to do both.

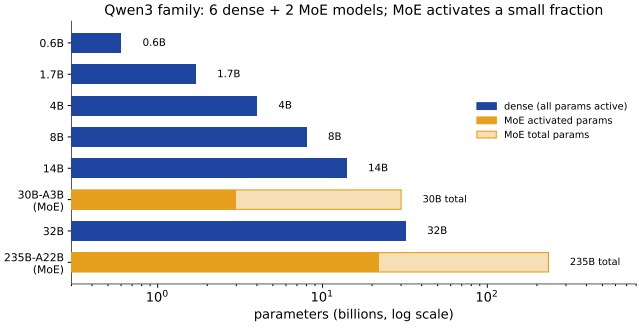


Outline

A family, not a model

Eight models from a phone-sized 0.6B to a 235B flagship – dense and MoE.

Eight models: dense 0.6B–32B, plus two MoE



6 dense models (0.6B, 1.7B, 4B, 8B, 14B, 32B) – every parameter runs on every token.

2 Mixture-of-Experts models:

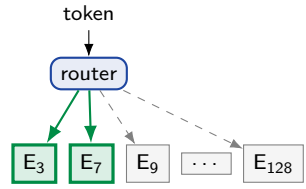
- **30B-A3B**: 30B total, only **3B active** per token.
- **235B-A22B** (flagship): 235B total, only **22B active**.

Same base architecture across the family; the MoE models just add the expert routing. One recipe, many sizes.

Mixture-of-Experts in one minute

Replace one big feed-forward block with **128 expert** blocks plus a small **router**. For each token the router picks the **top 8** experts; the rest stay idle.

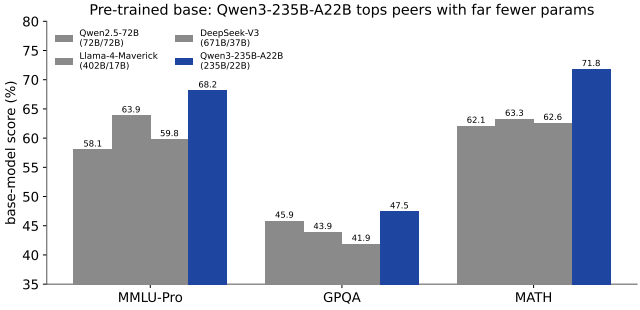
- You *store* 235B parameters but only *compute* with 22B per token.
- Capacity of a huge model, inference cost of a small one.
- Qwen3-MoE: fine-grained experts, **no** shared expert, global-batch load-balancing to push experts to specialize.



8 of 128 experts fire (green); the rest are skipped.

Many experts, few active – that is how a 235B model can serve at 22B-model cost.

The MoE payoff, in base-model scores



Before any alignment, the **pre-trained base** of Qwen3-235B-A22B already tops strong open peers:

- Beats **DeepSeek-V3-Base** on 14 of 15 benchmarks with $\approx$**1/3** the total and **2/3** the active parameters.
- Beats **Llama-4-Maverick** (about 2$\times$ the parameters) on most tasks.

Fewer active parameters, higher scores – the MoE design pays off in both quality and inference cost.

Where the raw ability comes from

36 trillion tokens, 119 languages, learned in three staged passes.

The pretraining data

Qwen3 is pre-trained on ≈**36 trillion tokens** across **119 languages and dialects** – twice the tokens and three times the languages of Qwen2.5 (which covered 29).

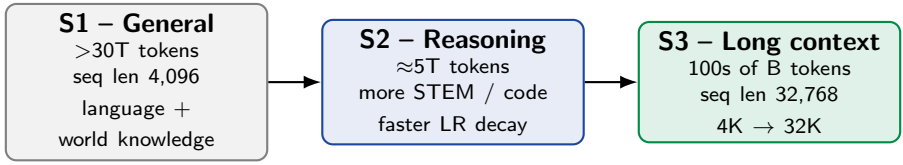
Where the extra data comes from:

- **PDF text**: Qwen2.5-VL reads scanned documents; Qwen2.5 cleans the text – trillions of new tokens.
- **Synthetic**: Qwen2.5-Math and Qwen2.5-Coder generate textbooks, Q&A, and code.
- Instance-level data mixing, tuned on small proxy models.

Models teaching models. Earlier Qwen models are the data pipeline for the next one – OCR, cleanup, and synthetic generation are all done by Qwen2.5 variants.

Expanding **29** → **119** languages is the headline multilingual jump over Qwen2.5.

Three pretraining stages



- **S1** builds broad competence; **S2** sharpens reasoning by up-weighting STEM, coding, and synthetic data; **S3** stretches the context window.
- Long context uses **RoPE base** $10^4 \rightarrow 10^6$ (ABF), plus **YARN + Dual Chunk Attention** for a $4\times$ length boost at inference.

Curriculum idea: learn to *speak* first, then to *reason*, then to *read long*.

One model, two modes – and a budget

Stop switching models. Switch a flag, and dial the thinking.

Unified thinking and non-thinking modes

A single Qwen3 model answers in **two** ways, chosen *at inference* by a flag in the prompt (or chat template):

- `/think` – emit a `<think>...</think>` chain, then the answer. Default.
- `/no_think` – leave the think block *empty*, answer immediately.

No more shipping a chat model *and* a separate reasoning model; no more routing requests between them.

Thinking mode

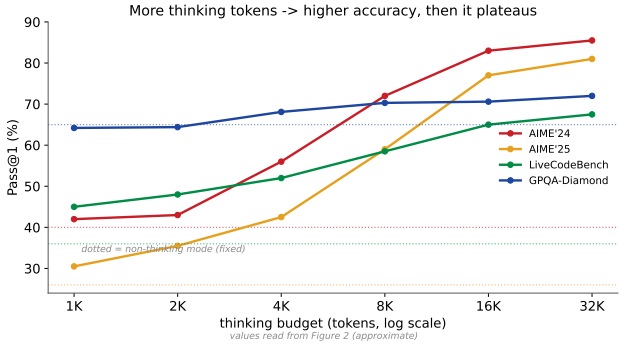
```
user: {q} /think
assistant: <think> reasoning
</think>
{answer}
```

Non-thinking mode

```
user: {q} /no_think
assistant: <think></think>
{answer}
```

The *empty* think block is deliberate – it keeps one output format for both modes, so the same weights serve both.

The thinking budget: a compute-vs-quality dial



Cap the thinking at N tokens. When the model hits the cap, it is **told to stop** ("*I have to give the solution now*") and answers from whatever reasoning it has so far.

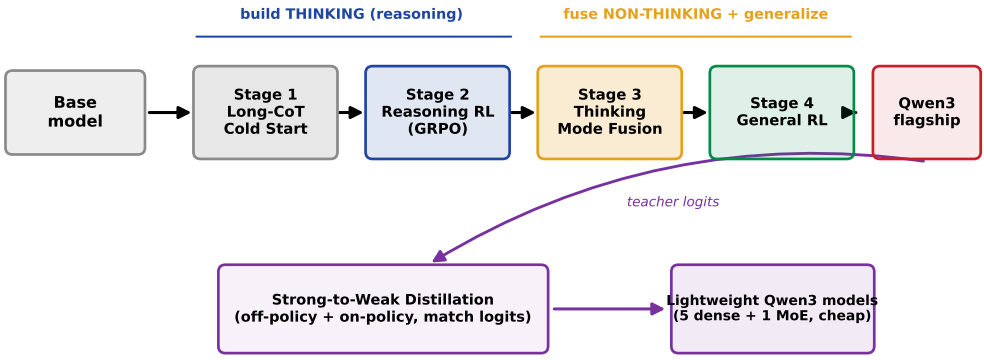
- More budget $\rightarrow$ steadily higher accuracy, **then a plateau**.
- Cheap questions: small budget. Hard questions: spend more.

This budget behavior was **never explicitly trained** – it *emerges* from the mode-fusion stage (coming up).

How the flagship learns both modes

Four stages: first build thinking, then fuse in non-thinking.

The four-stage post-training pipeline



- **Stages 1–2** grow reasoning (thinking) ability; **stages 3–4** add the non-thinking mode and generalize.
- Only the **flagship** models run all four stages. Small models take a cheaper route (distillation) – next section.

Stages 1–2: build the thinking

Stage 1 – Long-CoT cold start.

- Curate hard, *verifiable* problems (math, code, logic, STEM).
- QwQ-32B generates long chains of thought; heavy filtering keeps only clean ones.
- Light SFT: *instil* reasoning patterns, do not over-optimize – leave room for RL.

Stage 2 – Reasoning RL.

- 3,995 query–verifier pairs; reward = is the answer correct.
- Optimized with **GRPO** (RL as in **[LLM-1]**), big batches, many rollouts.
- One stable run lifts **AIME'24 from 70.1 → 85.1**.

Cold start gives the model the *habit* of reasoning; RL makes the reasoning actually *correct*. Verifiable rewards mean no reward model to hack.

Stages 3–4: fuse non-thinking, then generalize

Stage 3 – Thinking Mode Fusion.

- Continue SFT on a *mix* of thinking and non-thinking data.
- The thinking data is **self-generated** by the Stage-2 model (rejection sampling) so its reasoning is not degraded.
- Teaches the `/think` vs `/no_think` switch – and unlocks the **thinking budget**.

Stage 4 – General RL.

- RL over **20+ task types**: instruction following, format, tool use / agents, RAG, preference.
- Three reward kinds: **rule-based**, model-based *with* reference, model-based *without* reference.

Honest trade-off: fusion + general RL boost versatility (mode switching 88.7 $\rightarrow$ 98.9) but *slightly* dent peak AIME / code scores. The team accepts it for a more usable model.

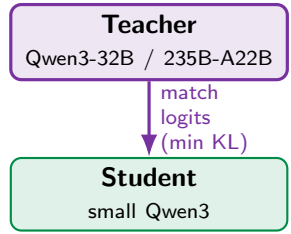
Making the small models cheaply

Do not RL every small model from scratch – let a big one teach it.

Big Qwen3 models teach the small ones

Running the full four-stage pipeline for every small model is wasteful. Instead, the 5 dense small models and the 30B-A3B MoE are built by **distillation** from Qwen3-32B / 235B-A22B:

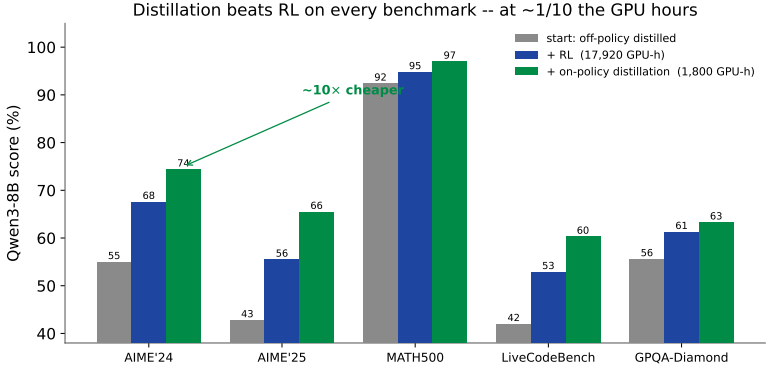
- **Off-policy**: student imitates teacher outputs in *both* modes – gets basic reasoning + mode switching.
- **On-policy**: student generates, then aligns its **logits** to the teacher's by minimizing **KL divergence**.



The teacher's full output distribution carries far more signal than a 0/1 "correct" reward – so the student learns faster *and* better.

Predict-first: distillation vs RL for a small model

For an 8B model starting from the *same* checkpoint: is it better to run reinforcement learning on it, or to distill from a bigger Qwen3? And which is cheaper?

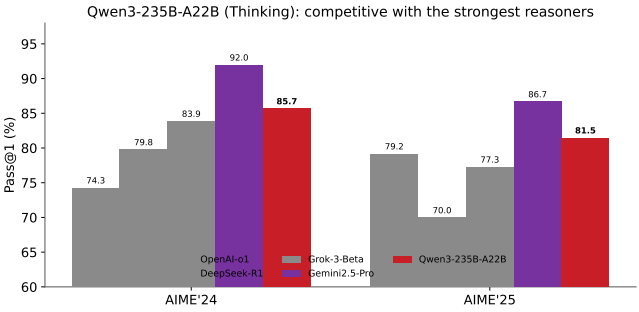


Distillation **wins on every benchmark** *and* costs ≈**1/10 the GPU hours** (1,800 vs 17,920). It even lifts **Pass@64** (more exploration), which RL does not. For small models, copy the big sibling.

Does it hold up?

Flagship vs the strongest open and closed reasoning models.

Competitive with the strongest reasoners



Qwen3-235B-A22B (Thinking):

- AIME'24 **85.7**, AIME'25 **81.5**, LiveCodeBench v5 **70.7**, BFCL v3 **70.8**.
- Beats DeepSeek-R1 on 17 of 23 benchmarks with far fewer parameters; competitive with OpenAI-o1 and Gemini2.5-Pro.

Open weights, **Apache 2.0** – state-of-the-art open reasoning that anyone can run.

Where Qwen3 lands

- **Both modes are strong.** The flagship is SOTA-among-open in *thinking* mode and beats DeepSeek-V3 / GPT-4o-class models in *non-thinking* mode.
- **Small models punch up.** Thanks to distillation, the lightweight models beat similarly-sized – and often larger – open models.
- **Multilingual.** 119 languages; strong on multilingual math and knowledge benchmarks.
- **Efficient.** MoE means flagship quality at a fraction of the active-parameter cost.

The practical win: one open model family that covers phone-to-server, thinks on demand, and lets you *pay for reasoning only when you need it*.

Recap

- **Family:** 6 dense (0.6B–32B) + 2 MoE (30B-A3B, 235B-A22B); MoE fires **8 of 128** experts, so it computes with a small fraction of its parameters.
- **Pretraining:** ≈36T tokens, 119 languages, three stages – general → reasoning → long context (up to 32K).
- **Headline idea:** *one* model with `/think` and `/no_think` modes plus a **thinking budget** – a compute-vs-quality dial that emerges from training, not a hand-coded rule.
- **Post-training:** 4 stages – long-CoT cold start → reasoning RL (GRPO, **[LLM-1]**) → thinking-mode fusion → general RL.
- **Small models:** strong-to-weak **distillation** beats RL-from-scratch – better *and* ≈10× cheaper.
- **Result:** competitive with o1 / R1 / Gemini2.5-Pro; open under Apache 2.0.

Takeaway: the frontier is no longer just "bigger" – it is *controllable inference-time compute* inside a single open model.