LLM 101: Efficiency, Compression & Distillation — 2025

Antreas Antoniou · AI Researcher · Lecturer · Founder, Axiotic AI

MSc/PhD Lecture — 2025/2026

> *"How do we build intelligence that is genuinely efficient — not just bigger?"*

Who Is Your Guide Today?

Antreas Antoniou — the short version:

- 🎓 **PhD in Machine Learning & Meta-Learning**, University of Edinburgh — Thesis: learning to learn from few examples
- 🔬 **Research Scientist**, Google · **Applied Scientist**, Amazon
- 🏛️ **Principal Scientist / Founder**, Axiotic AI — open-science research beyond the scaling paradigm

Why this lecture is personal:

> *I didn't just study this topic. I founded a company because I believe the future of AI is* **small, efficient, and local***.*

The question that drives everything:

> **How do we build intelligence that runs on fruit, not on a power station?**

The answer requires understanding what we're building — and then learning to make it radically more efficient.

The Banana Slide 🍌

Your brain weighs ~1.4 kg. Runs on **~20 watts** — roughly **3 bananas per day**.

A frontier AI training cluster runs on **10,000+ watts** — roughly **1,500 bananas per day**. And yet — your brain generalises from a handful of examples, adapts in real-time, reasons about novel situations, and does it all *on fruit*.

System	Power	Equivalent
Human brain	**~20 W**	~3 bananas/day
Frontier AI cluster	**~10,000 W**	~1,500 bananas/day

> 🔑 **The gap is not computational power. The gap is algorithmic intelligence.** *We're burning 500× more energy and still can't match the brain's flexibility.*

1

LLM 101

The Transformer, Training & Fine-Tuning

Before we can compress something, we need to understand what we're compressing.

Lecture Roadmap

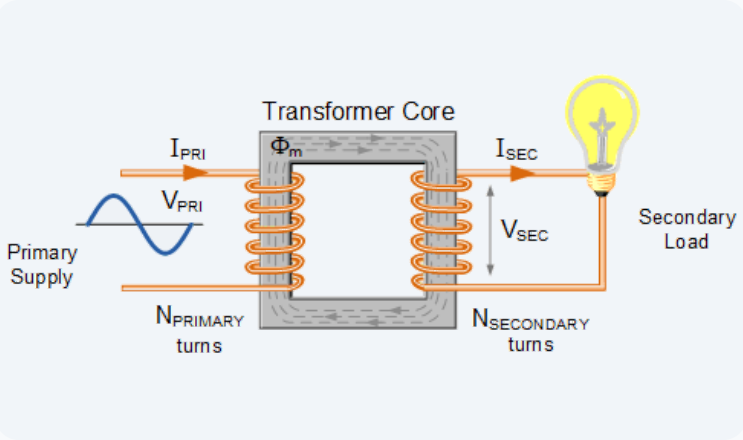
Section	Focus	Slides
1. LLM 101 🥾	Transformer architecture, training, fine-tuning	4–21
2. Compression & Distillation 🗜️	Pruning, quantization, distillation, efficiency	22–39
3. Smarter, Not Bigger 🧠	Local agents, intelligence per watt, the future	40–48
4. Resources & Big Picture 🗺️	Tools, papers, what to do next	49–54

> *At every stage I'll tell you where we are in the story. You'll never be lost.*

The Transformer — Not That Kind

When someone says "transformer" in AI, they do **not** mean:

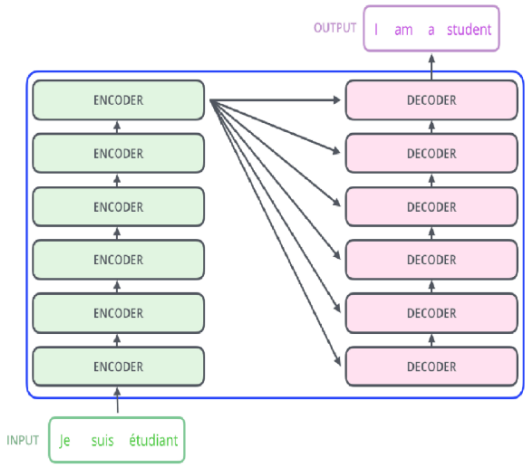
- ❌ An electrical transformer (coils and voltage) — *left*
- ❌ Optimus Prime (cool, but no) — *centre*
- ❌ Other mechanical gear — *right*

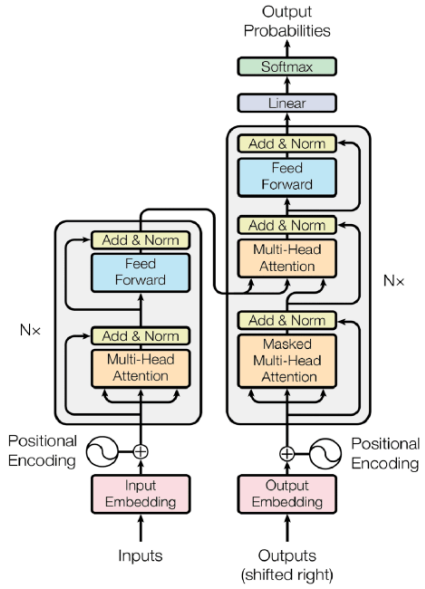


They mean a specific **neural network architecture** invented at Google in 2017 that has since dominated *every modality, every domain, and every task* in ML.

> *"Attention Is All You Need" — Vaswani et al., 2017 (arXiv:1706.03762)*

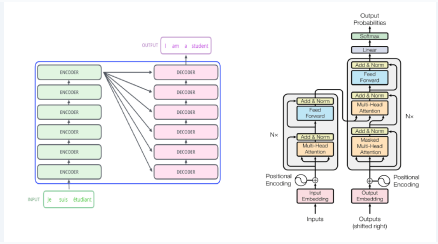
The Transformer Architecture





The Transformer Block: A Layered Sandwich

One block = Attention + FFN + Normalization + Skip Connections, repeated N times.



The original (2017) was

encoder-decoder

for machine translation. Modern LLMs use

decoder-only

— the task is next-token prediction on a single sequence.

> **The LEGO analogy:** *A few types of bricks — Token Embeddings, Positional Embeddings, Self-Attention, Feed-Forward, LayerNorm + Skip. Stack them. Same recipe, over and over. Simple rules, complex emergent results.*

The 2025 Transformer Recipe

Every major model (Llama 3, Gemma 2, Mistral, DeepSeek, Qwen) converged on:

① Tokenize — BPE or SentencePiece (vocabulary ~32K–128K tokens)

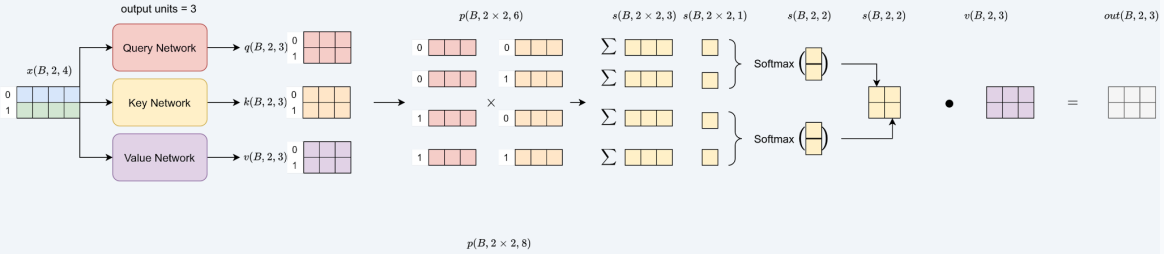
② Each transformer block:

- **RMSNorm** (not LayerNorm) — pre-norm: `x / sqrt(mean(x²) + ε) × γ` — ~15% fewer ops (Zhang & Sennrich, 2019)
- **Grouped Query Attention (GQA)** — reduces KV cache by sharing K,V heads across Q groups (Ainslie et al., 2023, arXiv:2305.13245)
- **SwiGLU** feed-forward — `SwiGLU(x) = Swish(xW₁) ⊙ (xV) × W₂` — consistently outperforms ReLU/GELU (Shazeer, 2020)
- **Residual connections** (unchanged from 2017 — still essential)

③ Autoregressive next-token prediction

> *Common misconception: "Transformers use LayerNorm and ReLU." No. Not since 2023.*

Self-Attention — The Core Mechanism



The exam hall analogy: For each question, peek at the classmates worth copying — history nerd for history, maths genius for maths.

Self-attention does exactly this — for each token:

- **Query (Q):** "What am I looking for?"
- **Key (K):** "What do I have to offer?"
- **Value (V):** "Here's my actual content"

$$\text{Attention}(Q,K,V) = \text{softmax}(QK^T/\sqrt{d\_k})V$$

Why √d_k? Without it, dot products grow with dimension, pushing softmax into saturation, killing gradients.

Multi-Head Attention (MHA): h parallel heads with different projections. Each learns different relationships (syntax, coreference, semantics).

Attention Variants — MHA → GQA → MQA

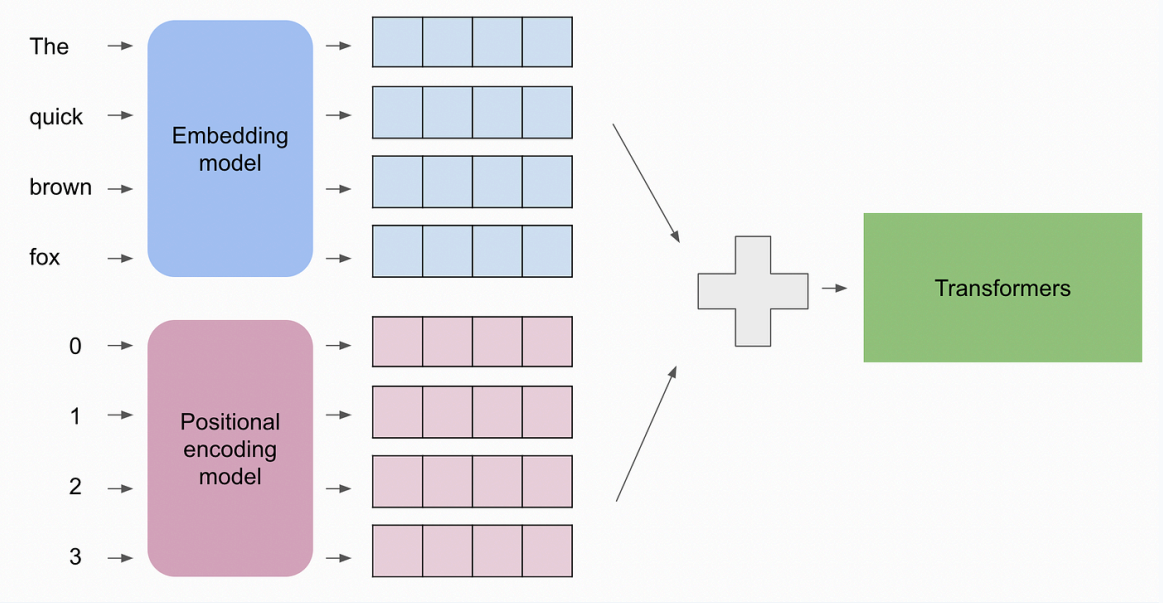
The KV cache problem: During generation, we cache K and V for all previous tokens. For a 70B model with 80 heads, 128 dim/head, 8K context: **~20 GB KV cache per sequence**. This is the serving bottleneck.

Variant	Year	K,V Heads	KV Cache	Used In
MHA (Multi-Head)	2017	h	Baseline	BERT, GPT-2/3
MQA (Multi-Query)	2019	**1**	**÷ h**	PaLM, Falcon
GQA (Grouped-Query)	2023	g (1<g<h)	**÷ (h/g)**	**Llama 2/3, Mistral, Gemma, DeepSeek**

GQA is the new standard. Llama 2 70B: 64 Q heads, 8 KV heads → 8× KV cache reduction. MHA is effectively deprecated for large-scale deployment.

FlashAttention (Dao et al., 2022, arXiv:2205.14135): Rewrites attention to work in GPU SRAM tiles. 2-4× speedup, linear memory, exact output. Default in PyTorch ≥2.0.

Positional Embeddings — The Evolution



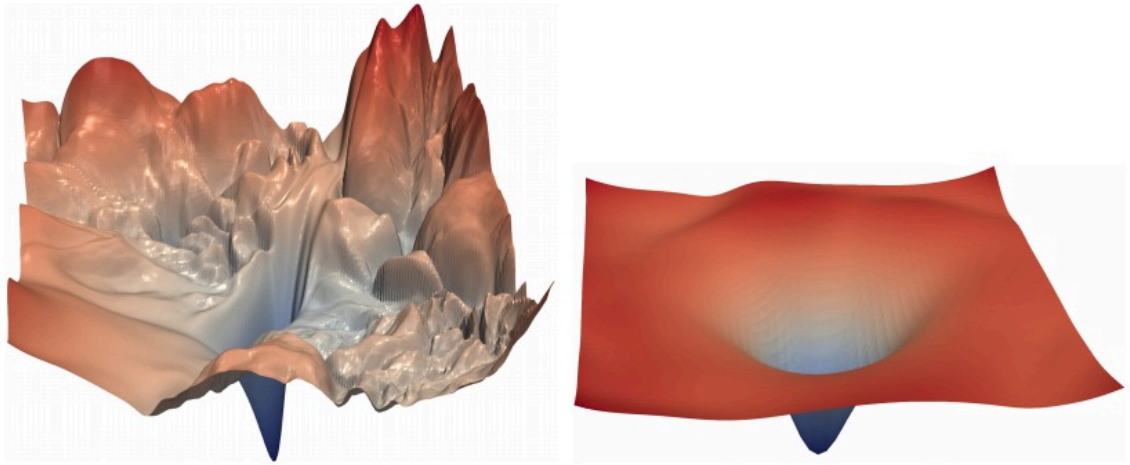
Transformers are **permutation-equivariant** by default — "Dog bites man" = "Man bites dog" without positional information.

Analogy: Positional embeddings are **seat numbers in a lecture hall**.

Method	Year	Status
Sinusoidal	2017	Original paper
Learned absolute	2018	Fixed max length
RoPE	2021	**De facto standard**
ALiBi	2022	Simple, no learned params
YaRN	2023	**128K+ from 4K training**

RoPE: Applies rotation matrices to Q and K. The dot product naturally encodes relative distance — relative position emerges from geometry.

Skip Connections & Why Transformers Work



(a) without skip connections (b) with skip connections

Figure 1: The loss surfaces of ResNet-56 with/without skip connections. The proposed filter normalization scheme is used to enable comparisons of sharpness/flatness between the two figures.

Skip connections — arguably the most important idea in deep learning:

`output = Layer(x) + x` — the identity shortcut

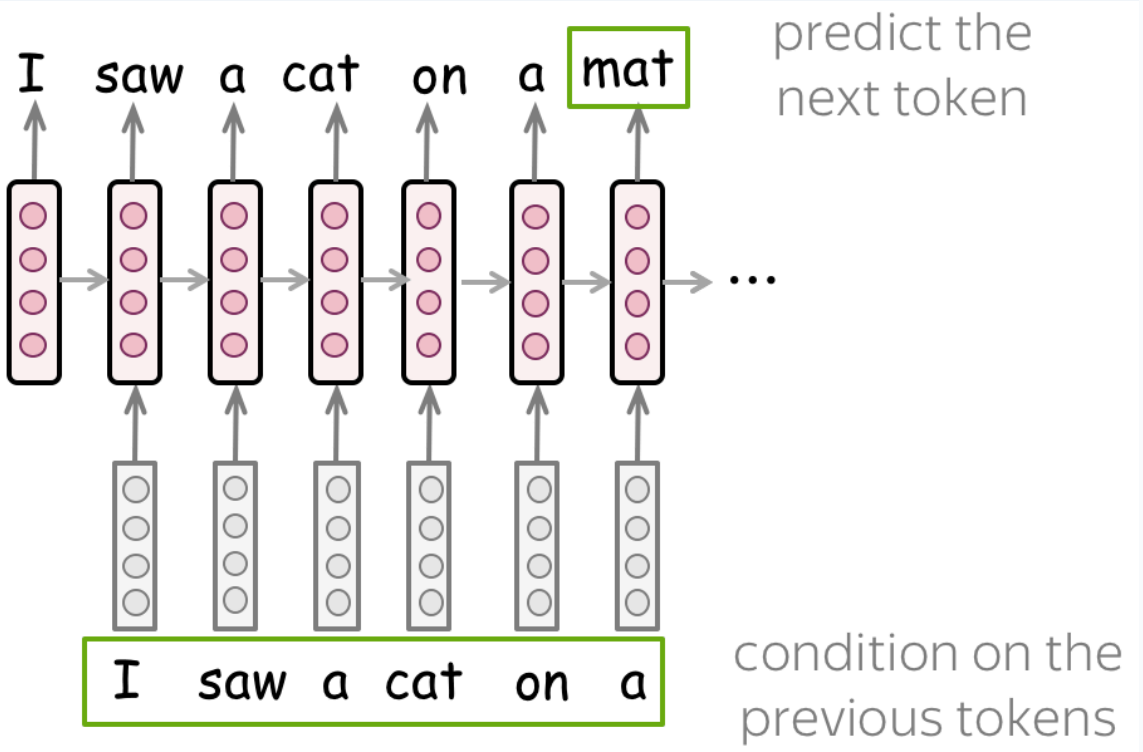
- Without: chaotic, jagged loss landscape with sharp local minima
- With: smooth, nearly convex — easy to optimise (Li et al., 2018, arXiv:1712.09913)

What makes Transformers uniquely powerful? (Antreas's opinion):

- **Relational learning** — attention computes pairwise interactions. Representations are relative, not absolute.
- **Learnable patterns** — the shape of attention is learned, not fixed.
- **Data-conditional computation** — functions depend on other tokens present.

> *"The Transformer is fundamentally a series of meta-learning networks. Learning to learn was underneath it all, all along!"*

Pretraining — Autoregressive Language Modelling



Analogy: Give a student millions of exam papers with the last answer always blanked. Their job: **predict the next word**, billions of times.

$$P(x_1,...,x_n) = \prod_i P(x_i \mid x_1,...,x_{i-1})$$

- **Loss:** Cross-entropy between predicted distribution and actual next token
- **Masked self-attention:** Each position only attends to positions ≤ itself
- **Training is parallel:** All positions computed simultaneously (teacher forcing)

The Chinchilla insight (2022): GPT-3 was dramatically undertrained. Modern: Llama 3 8B trained on **15T tokens**.

> 🔑 *Next-token prediction forces the model to build a rich internal world model — sufficient to learn grammar, facts, reasoning, coding, and common sense.*

Pretraining — Diffusion Language Models

The challenger to autoregressive dominance:

	Autoregressive	Diffusion
How	Tokens one by one, L→R	Denoise all tokens simultaneously
Speed	Slow (sequential)	Fast (parallel)
Quality	SotA (2025)	Catching up fast
Examples	GPT, Llama, Claude	Mercury (Inception Labs)

Mercury (Inception Labs, 2025, arXiv:2502.09992):

- First diffusion LM competitive with AR models on standard benchmarks
- **3× faster generation** — parallel vs sequential
- Strong for structured outputs where global coherence matters

> *This is the first credible alternative to autoregressive generation in 8 years. Pay attention.*

Architecture Families — Dense → MoE → SSM → Hybrid

Family	Key Idea	Examples	Tradeoff
Dense Transformer	Full attention, all params active	GPT-4, Llama 3, Gemma 2	Maximum quality, maximum cost
MoE	Route tokens to specialised sub-networks	Mixtral, **DeepSeek-V3**	Capacity » compute cost
SSM (Mamba)	Linear-time recurrence, no attention	Mamba, S4, RWKV	Fast inference, less proven at scale
Hybrid	Attention + recurrence + memory	Jamba, Griffin	Best of both worlds? Under research

DeepSeek-V3 (Dec 2024): 671B total params, only **37B active per token**. Trained for **~$5.57M**. Competitive with GPT-4o.

Mamba (Gu & Dao, 2023, arXiv:2312.00752): O(N) time and memory. Strength: long-context streaming. Weakness: in-context learning. *SSMs are not a Transformer replacement yet — they are a complement.*

Reasoning Models — Test-Time Compute as New Scaling Axis

The breakthrough idea: Instead of only scaling *training* compute, scale *inference* compute.

OpenAI o1/o3 (2024-2025)

- Extended chain-of-thought reasoning before answering, trained with RL
- Can "think harder" on difficult problems by generating more tokens
- o3: >90% on ARC-AGI, ~96% on AIME math competition

DeepSeek-R1 (Jan 2025, arXiv:2501.12948)

- Open-weight 671B MoE reasoning model
- **Pure RL (GRPO)** on base model produces emergent CoT reasoning *without* supervised CoT data
- Matches o1-level performance on math and coding — **open-weight + recipe published**

The test-time scaling law: More reasoning tokens → better answers, following a smooth scaling curve. A 7B model thinking for 2000 tokens can outperform a 70B model thinking for 100.

> 🔑 *You don't need a bigger model — you need a model that* **thinks longer**.

Fine-Tuning — SFT → RLHF → DPO

The modern fine-tuning pipeline (2025):

```
Base model (pretrained) → [SFT] → [Preference Alignment] → Aligned model (deployed)
```

SFT (Supervised Fine-Tuning)

Train on (instruction, response) pairs. Standard cross-entropy. Teaches format/style. *"What to say."*

RLHF (Ouyang et al., 2022)

Train a reward model on human preferences. Optimise policy with PPO. Complex: 4 models in memory. *"What's good."*

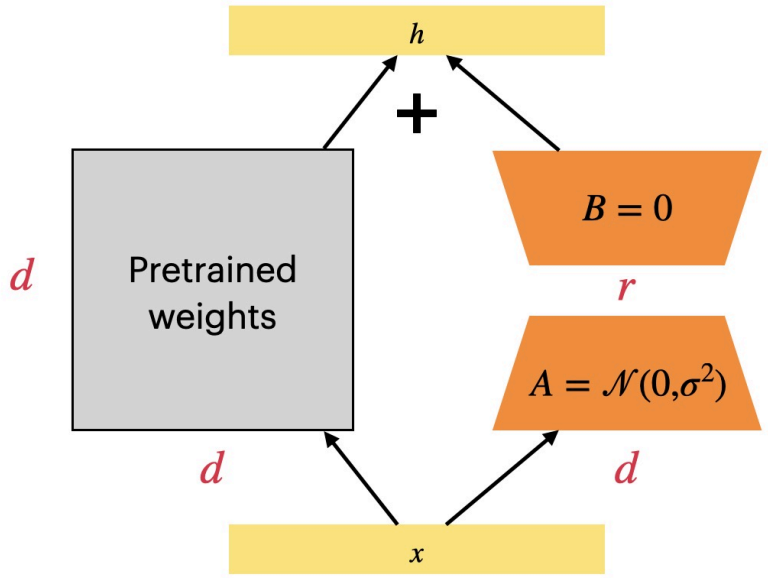
DPO (Rafailov et al., 2023, arXiv:2305.18290)

Closed-form relationship between optimal RLHF policy and reward function. Eliminates reward model entirely. Simpler, more stable. **Now the default** for open-source alignment.

Variants: ORPO (Hong, 2024) — SFT + alignment in one step. SimPO (Meng, 2024) — reference-model-free. KTO (Ethayarajh, 2024) — unpaired preferences.

> ✋ **Think:** *SFT teaches "what to say." RLHF/DPO teaches "what's good" — capturing helpfulness, harmlessness, honesty.*

PEFT vs Full Fine-Tuning



Full Fine-Tuning (FFT):

- ✅ Updates **all** model parameters — best quality for deep domain shifts
- ❌ For 70B in bf16: ~140 GB model + ~420 GB optimizer states (AdamW)

LoRA (Hu et al., 2021, arXiv:2106.09685):

- Freeze pretrained weights. Add: `W' = W + BA` where r ≪ d
- Trainable params ≈ 0.1–1% of total
- *Analogy:* Instead of repainting a building, add a thin film to each window

QLoRA (Dettmers et al., 2023, arXiv:2305.14314):

- Base model in **4-bit NormalFloat** + LoRA adapters in bf16
- Fine-tune 65B on a single 48GB GPU — **democratised LLM fine-tuning**

> **Practical guidance:** *Have H100s? → FFT wins. Single GPU? → QLoRA is remarkable. Many task variants? → LoRA adapters.*

On Training the LLM Beast

There are three main directions for training LLMs efficiently:

1. Single GPU Training

- Best for models that fit comfortably in one GPU's memory
- Standard DataLoader + optimizer loop, no parallelism overhead
- Limited to models ~7B on 80GB A100; use quantization to push further

2. Multi-GPU Data-Parallel Training

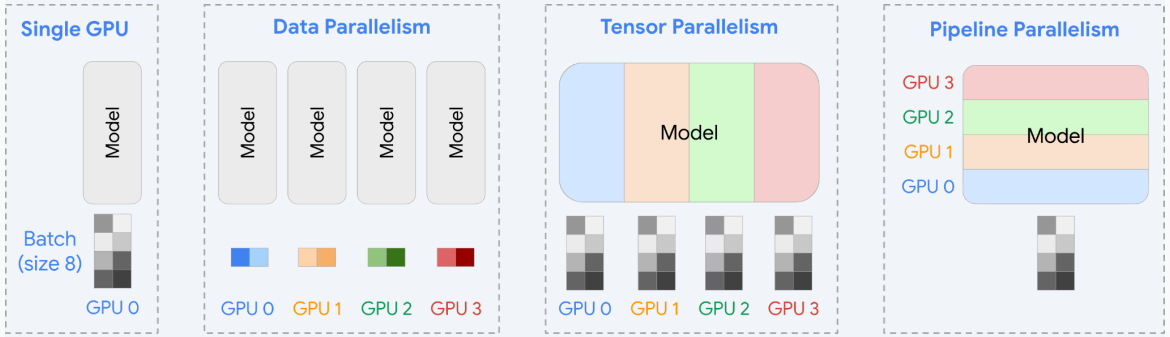
- Each GPU holds a full copy of the model
- Different data batches processed in parallel, gradients averaged
- **Does not work** for models requiring more than 1 GPU's memory for forward + backward pass

3. FSDP — Fully Sharded Data Parallelism

- Shards model parameters, gradients, *and* optimizer states across GPUs
- Enables training models that are too large to fit on any single GPU
- Each GPU holds only its shard; parameters are gathered on-demand during compute

> 🔑 *FSDP is the key technique for fine-tuning frontier-scale models without an ocean of memory — it's what makes* **Llama 3 405B fine-tuning** *practical.*

On FSDP — Fully Sharded Data Parallelism



What is FSDP?

- Memory-efficient distributed training for large models
- Shards parameters, gradients, and optimizer states across all GPUs
- Reduces per-GPU memory — enables larger batch sizes and training stability

FSDP Modes

- **Full Parameter Sharding:** Weights, gradients, and optimizer states all sharded
- **Mixed Precision (bf16/fp16):** Reduces memory while maintaining stability
- **Activation Checkpointing:** Saves memory by recomputing activations during backprop
- **Auto Wrapping:** Automatically applies sharding to transformer blocks

> **Note:** *Optimal hyperparameters depend on hardware, model size, task, and data. Tune per workload.*

Transformer Timeline — 2017 to 2025

Year	Milestone	Scale
2017	"Attention Is All You Need" at Google	65M params
2018	GPT-1 (OpenAI adopts Transformer)	117M
2018	BERT (bidirectional) — dominates NLP overnight	340M
2019	GPT-2 — first "scary good" generation	1.5B
2020	GPT-3 — era of scaling begins	175B
2022	Chinchilla — reveals models were undertrained	70B (better)
2022	ChatGPT — the public awakening	—
2023	GPT-4, Llama, open-source explosion	~1.8T (MoE)
2024	Reasoning models (o1), Llama 3.1, MoE	405B dense
2025	DeepSeek-V3/R1, compression revolution	671B MoE (37B active)

The Transformer has dominated every modality, domain, and task for 8 years. No architecture in ML history has had this kind of run.

2 — Model Efficiency, Compression & Distillation

Making Models Smaller, Faster, and Smarter

> *Something has to change. The scaling recipe works — but at what cost?*

The transition from "just make it bigger" to "make it smarter" is the central challenge of AI in 2025.

- Section structure: Efficiency taxonomy → Pruning → Quantization → Distillation → Inference efficiency
- Each technique reduces model cost while preserving capability
- They **compose** — a pruned, quantized, distilled model can be 100× smaller with <5% quality loss

Three Kinds of Efficiency

🖥️ Computational	💡 Idea	🔬 Research
Efficient architectures	Simplicity	Ease of access & use
Efficient code	Composability	Ease of modification
Efficient hardware	Cognitive complexity	Ease of evaluation

The best efficiency gains come from better ideas, not just better hardware. *A breakthrough in idea efficiency (FlashAttention, MoE routing, the Transformer itself) yields orders-of-magnitude computational gains downstream.*

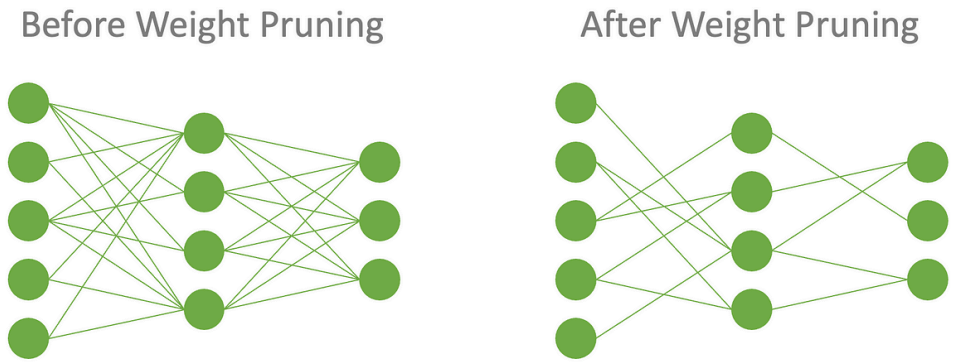
The Compression Toolkit — Overview

Tool	What It Does	Typical Result	Analogy
Pruning 🔪	Remove low-importance connections	2–10× sparser	Sculptor removing marble to reveal the statue
Quantization 📐	Reduce numerical precision	2–8× smaller	Approximating π as 3.14 instead of 3.14159265...
Distillation 🧪	Train small model to mimic large	10–100× smaller	A wise teacher training a bright student
Architecture 🏗️	Build efficiency in (MoE, SSM)	Variable	Purpose-built, not retrofitted

The power move: A quantized, pruned, distilled model on an efficient architecture can be **100× smaller** with <5% quality loss.

The modern pipeline: Train big → Distill small → Quantize → Deploy

Pruning — SparseGPT, Wanda, and 2:4 Sparsity



Core idea: Not all connections matter equally. Find and remove the unimportant ones.

The Lottery Ticket Hypothesis (Frankle & Carlin, 2018, arXiv:1803.03635): Within a large network exists a small subnetwork that performs equally well — but you must train the big one first to find it.

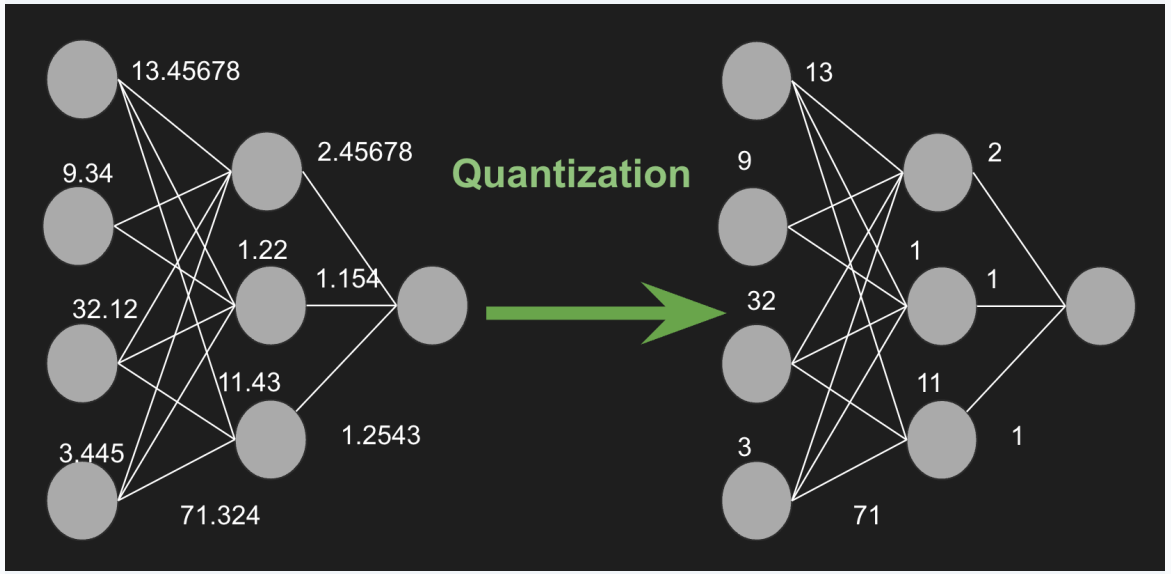
SparseGPT (2023, arXiv:2301.00774): One-shot pruning — no retraining. Prunes to **50-60% sparsity** with minimal perplexity increase.

Wanda (2024): Prune by **|weight| × |activation|** — comparable to SparseGPT, no Hessian needed.

2:4 Semi-Structured Sparsity (NVIDIA): In every 4 weights, exactly 2 must be zero. **Hardware-accelerated** on Ampere/Hopper — 2× speedup. The only sparsity with real hardware support.

> *Honest assessment: unstructured sparsity is hard to accelerate. 2:4 structured sparsity is the pragmatic path.*

Quantization — The 2025 Landscape



Core idea: Use fewer bits per weight. Dramatic compression for minimal quality loss.

Precision	Bits	Size (70B)	Quality	Sweet Spot?
FP32	32	280 GB	Baseline	Training only
FP16/BF16	16	140 GB	~Same	Standard training
INT8	8	70 GB	~99%	Post-training quant
INT4	**4**	**35 GB**	**~95-98%**	✅ **Inference sweet spot**
BitNet 1.58	1.58	~14 GB	~Same*	Research frontier

PTQ (Post-Training): Quantize after training. Fast. **QAT** (Quantization-Aware Training): Train with quantization in loop. Better quality, higher cost.

Quantization Methods — The Ecosystem

Method	Bits	Type	Key Innovation	Best For
bitsandbytes (NF4)	4	PTQ	NormalFloat + double quant	QLoRA fine-tuning
GPTQ	3-4	PTQ	Hessian-based layer-wise quant	GPU inference, large ecosystem
AWQ	4	PTQ	Protects activation-salient channels	Better quality than GPTQ
EXL2	2-8 mixed	PTQ	Per-layer flexible bit-width	Max quality for size budget
GGUF (llama.cpp)	2-8	PTQ	CPU-optimised, mixed quant	Local/CPU inference
BitNet b1.58	1.58	QAT	Ternary {-1,0,1} from scratch	Research frontier

AWQ — Why Activation Awareness Matters

AWQ (Lin et al., 2024, arXiv:2306.00978) — the state of the art for PTQ in 2025.

The Core Insight

- Not all weights are equally important
- **~1% of weight channels** carry disproportionate information
- These "salient" channels correspond to **large activation magnitudes** (not large weight magnitudes!)
- Naive quantization destroys these critical channels → quality collapse

The Solution

- Identify salient channels by measuring activation magnitudes on a small calibration set
- Apply **per-channel scaling** to protect salient channels before quantization
- Scale factors absorbed into adjacent layers — **zero overhead at inference**

> **Why this is elegant:** *Activation statistics (not weight statistics) determine quantization sensitivity. Now the default quantization method in vLLM and most serving frameworks.*

BitNet b1.58 — The Ternary Future?

BitNet b1.58 (Ma et al., 2024, arXiv:2402.17764)

Every weight is one of three values: **{-1, 0, 1}** — that's $\log_2(3) \approx 1.58$ bits.

How: QAT from scratch. Weights quantized via absmean: `w' = Round(w / mean(|w|)) ∈ {-1, 0, 1}`. Activations quantized to INT8 per-token.

Key Results

- At 3B params: **matches FP16 Transformer perplexity** at same model size and training tokens
- Dramatic improvements in latency, memory, throughput, and energy

The Radical Implication

- Matrix multiplication becomes **integer addition** — no floating point hardware needed
- Opens the door for **custom silicon** optimised for ternary ops
- Could enable LLM inference on hardware without an FPU

> ✋ **Think:** *If BitNet works at scale, we don't need FP hardware for AI. What does that imply for chip design? For putting AI in a wristwatch?*

The GGUF Revolution — LLMs for Everyone

GGUF (GPT-Generated Unified Format) — the file format that democratised local AI. Created by the **llama.cpp** project (Georgi Gerganov, 2023-2024).

- **CPU-first inference** — runs on x86, ARM, Apple Silicon, RISC-V
- Mixed quantization: different bit-widths per tensor type
- Quality tiers: Q2_K (tiny, lossy) → **Q4_K_M (sweet spot)** → Q8_0 (near-lossless)

The Ecosystem Built on GGUF

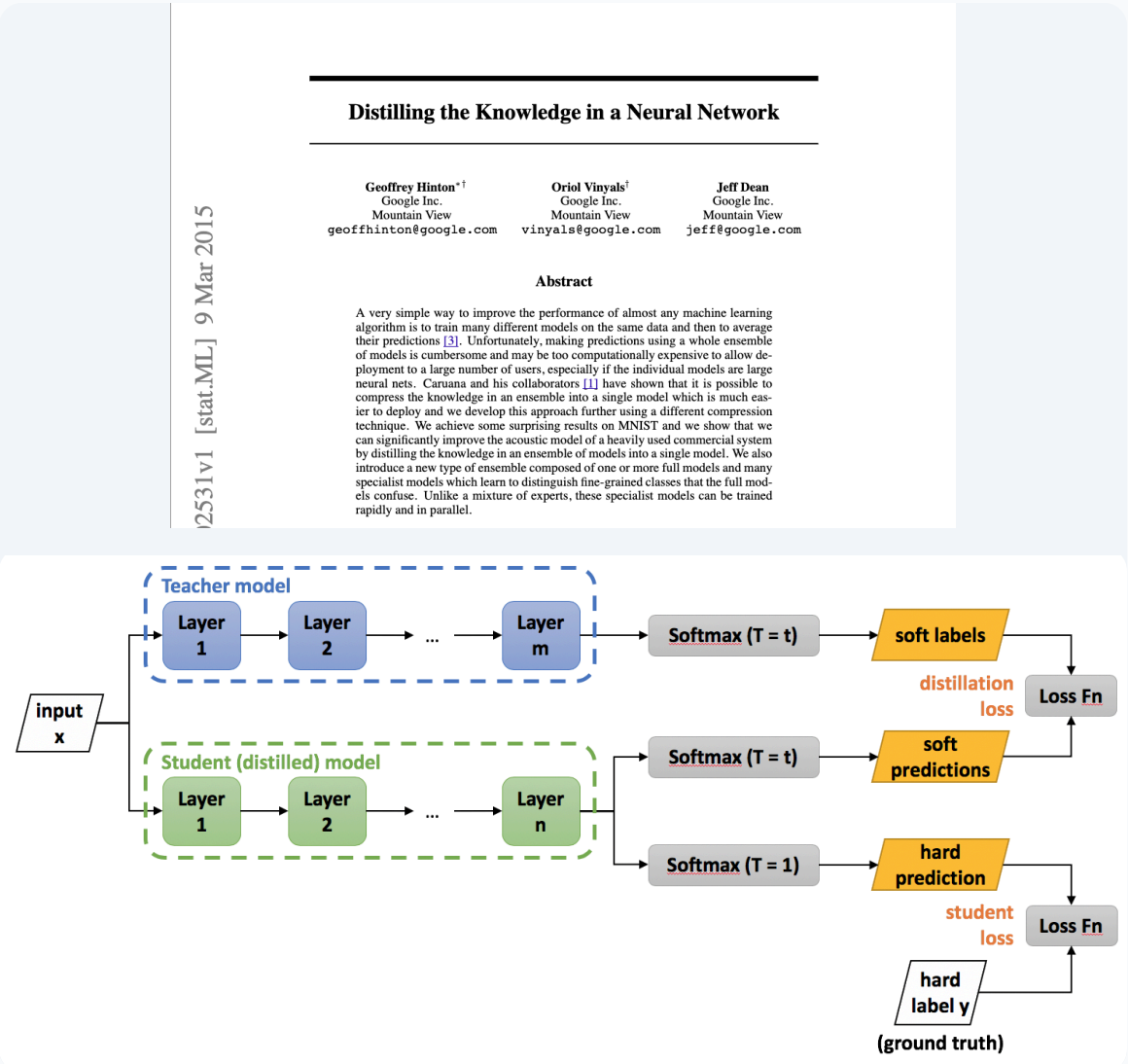
- **Ollama** — one-command local LLM (`ollama run llama3.2`)
- **LM Studio** — GUI for model exploration
- **Open WebUI** — ChatGPT-like interface, fully local

The Democratisation Story

- 2023: Running a 7B model required ≥$1,000 GPU
- 2025: GGUF Q4 runs 7B on a **MacBook Air**, a **Raspberry Pi 5**, or a **phone**

> 🔑 *The real benchmark isn't MMLU — it's "can a grad student run this on their laptop?"*

Distillation — Foundations



Distilling the Knowledge in a Neural Network

Geoffrey Hinton*† Google Inc. Mountain View geoffhinton@google.com
Oriol Vinyals† Google Inc. Mountain View vinyals@google.com
Jeff Dean Google Inc. Mountain View jeff@google.com

Abstract

A very simple way to improve the performance of almost any machine learning algorithm is to train many different models on the same data and then to average their predictions [3]. Unfortunately, making predictions using a whole ensemble of models is cumbersome and may be too computationally expensive to allow deployment to a large number of users, especially if the individual models are large neural nets. Caruana and his collaborators [1] have shown that it is possible to compress the knowledge in an ensemble into a single model which is much easier to deploy and we develop this approach further using a different compression technique. We achieve some surprising results on MNIST and we show that we can significantly improve the acoustic model of a heavily used commercial system by distilling the knowledge in an ensemble of models into a single model. We also introduce a new type of ensemble composed of one or more full models and many specialist models which learn to distinguish fine-grained classes that the full models confuse. Unlike a mixture of experts, these specialist models can be trained rapidly and in parallel.

"Distilling the Knowledge in a Neural Network" — Hinton, Vinyals, Dean (2015, arXiv:1503.02531)

Core idea: A smaller "student" learns from a larger "teacher's" **soft predictions**.

Why soft labels beat hard labels:

- Hard label: "this is a cat" — one bit of information
- Soft label: "90% cat, 5% dog, 3% tiger..." — rich relational structure
- The teacher's distribution over *wrong* answers reveals similarity — this "dark knowledge" is where the magic lives

The distillation loss:

$$L = \alpha \cdot CE(y, \sigma(z_s)) + (1-\alpha) \cdot T^2 \cdot KL(\sigma(z_t/T) \parallel \sigma(z_s/T))$$

Where T = temperature (higher → softer distributions → more dark knowledge exposed)

The Temperature Trick & The Conundrum

The Temperature Trick:

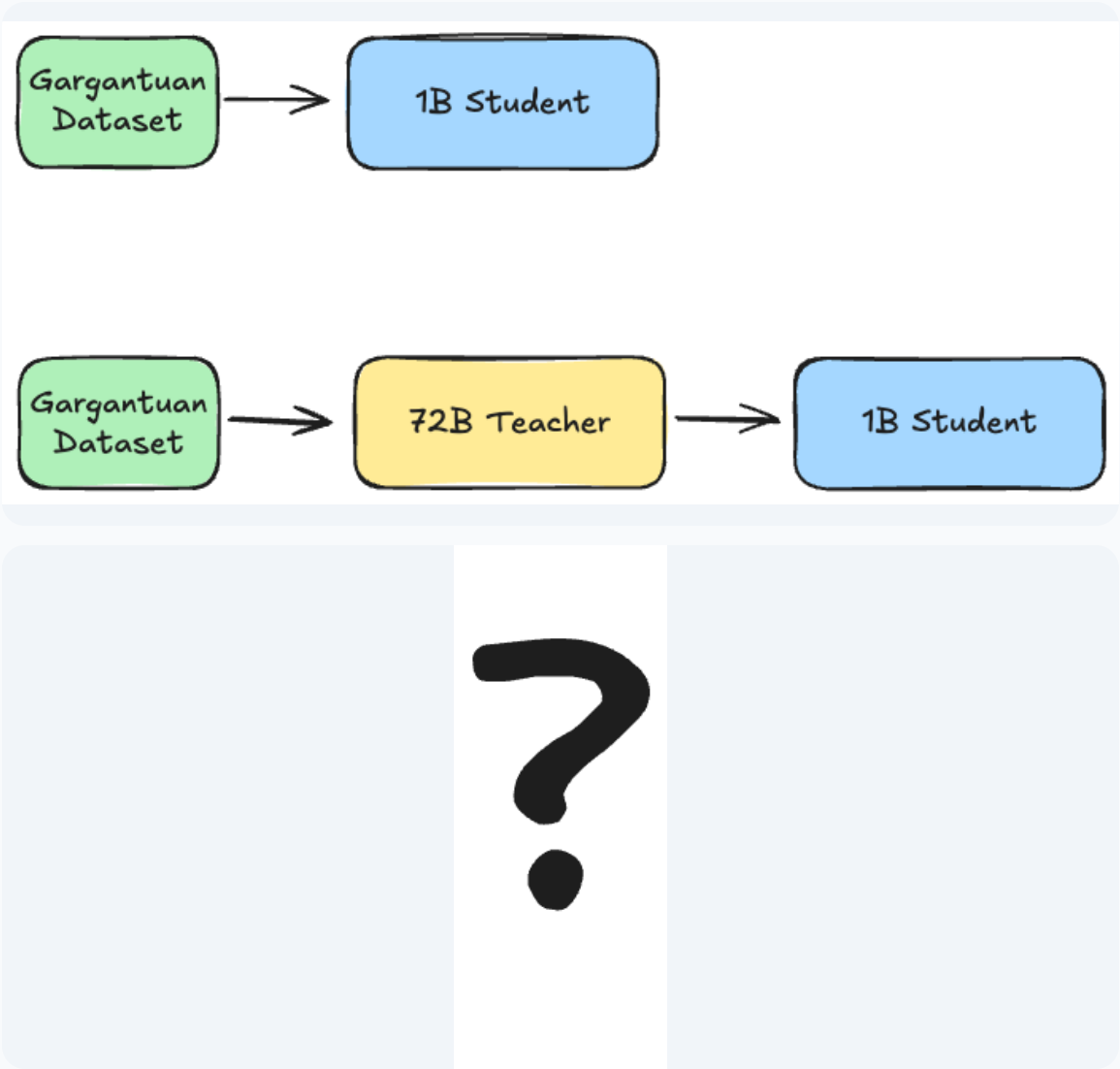
Raw teacher output: Cat 99.2%, Dog 0.7%, Fox 0.1% — too sharp.

With T > 1: Cat 70%, Dog 20%, Fox 10% — now student learns: *"It's a cat, but it kind of looks like a dog."*

The Conundrum: Two paths to a 1B student:

- **Path A:** Train 1B directly on a giant dataset
- **Path B:** Train a 72B teacher first, then distill to 1B

Path B wins — often dramatically.



Distillation Types — 2025 Taxonomy

Classical:

Type	What's Transferred	Reference
Knowledge Distillation (KD)	Soft labels (output logits)	Hinton et al. (2015)
Progressive Distillation	Chain: teacher → medium → small	Furlanello et al. (2018)
Layer-wise Distillation	Hidden representations per layer	Romero et al. (2015) FitNets
Task-Specific Distillation	Domain fine-tuned knowledge	Sun et al. (2019) TinyBERT

Modern (2024-2025):

Type	What's Transferred	Key Example
Reasoning Distillation	Chain-of-thought traces	DeepSeek-R1 → R1-Distill-7B
Synthetic Data	Teacher-generated training data	GPT-4 → Phi-3, Gemini → Gemma
Constitutional AI / RLAIF	AI-generated preference labels	Claude's Constitutional AI
Behaviour Cloning	Input-output pairs from teacher API	Orca, Vicuna, Alpaca

> 🔑 *The boundary between "distillation" and "synthetic data generation" has blurred. When GPT-4 generates training data for Phi-3, that's functionally distillation.*

DeepSeek-R1 — Distilling Reasoning Itself

The most important distillation result of this era (DeepSeek-AI, Jan 2025, arXiv:2501.12948)

The Recipe

- Train DeepSeek-R1 (671B MoE) with pure RL (GRPO) to reason via chain-of-thought
- Generate **~800K reasoning traces** across maths, coding, science, logic
- Fine-tune smaller base models (Qwen-2.5, Llama-3) on these traces — **pure SFT, no RL on student**

Model	Params	AIME 2024	Codeforces	Runs On
OpenAI o1-mini	Unknown	63.6%	1820 Elo	API only
DeepSeek-R1	671B (37B active)	79.8%	2029 Elo	~4× H100
R1-Distill-Qwen-32B	**32B**	**72.6%**	**1691 Elo**	**1× H100**
R1-Distill-Qwen-7B	**7B**	**55.5%**	**1189 Elo**	**Laptop**
R1-Distill-Qwen-1.5B	**1.5B**	**28.9%**	—	**Phone**

> 🔑 *Proves* **reasoning itself is distillable**. *A 7B model on a laptop doing competition mathematics. The teacher's CoT is a structured curriculum — the student learns how to think.*

Synthetic Data as Distillation — The New Pipeline

The pattern: Teacher generates data → Student trains on it → Student rivals models 10-50× its size

Student	Params	Teacher	Key Result
Phi-3-mini	3.8B	GPT-4 + web	Rivals Mixtral-8×7B (46.7B) on reasoning
Orca 2	7B/13B	GPT-4	Strong step-by-step reasoning via explanation tuning
Gemma 2	2B/9B/27B	Gemini	2B competitive with models 5× larger
Llama 3.2	1B/3B	Llama 3.1 405B	SotA sub-3B via pruning + distillation
Qwen 2.5	0.5B–72B	Internal	Competitive across all sizes

> *"This is distillation's killer app: not matching logits, but generating wisdom."*

The uncomfortable implication: The best small models are all distilled from proprietary large models. Exception: DeepSeek-R1 — open-weight teacher, open recipe.

PRMs and Constitutional AI

Process Reward Models (PRMs)

- Instead of judging only the final answer (outcome reward), PRMs score **each intermediate reasoning step**
- Dense, step-level supervision → more informative training signal
- Enables self-improving loops: model generates reasoning → PRM scores steps → model improves
- Key to making reasoning models (o1, R1) work reliably

Constitutional AI / RLAIF (Anthropic)

- Replace expensive human preference labelling with **AI-generated feedback**
- Define a "constitution" (principles the model should follow)
- The AI critiques and revises its own outputs against these principles
- Generates preference pairs automatically → train with DPO/RLHF
- Scales supervision without scaling human annotation costs

> **Connection to distillation:** *Both PRMs and Constitutional AI are forms of* **automated knowledge transfer** *— the system generates its own training signal, philosophically distillation from encoded principles.*

Speculative Decoding — Free Speed, Mathematically Exact

The most elegant efficiency technique in modern inference.

The problem: Autoregressive decoding is **memory-bandwidth bound**. The GPU loads full model weights for each token but does very little computation. Hardware is >90% idle.

The solution (Leviathan et al., 2023, arXiv:2211.17192):

- Small **draft** model (e.g., 1B) generates K candidate tokens — **fast**
- Large **target** model (e.g., 70B) scores all K tokens in **ONE forward pass**
- Accept all tokens up to first rejection; resample at rejection point
- **Output distribution is EXACTLY the target model's** — mathematically proven

> **Why it works:** *Verifying K tokens takes ~the same time as generating 1. If draft acceptance rate is ~70-80%:* **2-3× wall-clock speedup for free**.

The distillation connection: A better draft model → higher acceptance rate → more speedup. Acceptance rate is a **quantitative measure of distillation quality**.

Used in production: Google (Gemini), Anthropic (Claude), most major providers.

The inference stack:

- FlashAttention v2/v3 — tiled SRAM attention, 2-4× speedup
- PagedAttention (vLLM) — virtual memory for KV cache, 2-4× more concurrent users
- Continuous batching — dynamic add/remove requests mid-generation

> **2025 minimum serving stack:** *vLLM + FlashAttention + PagedAttention + continuous batching + speculative decoding. This is not optional.*

FlashAttention & PagedAttention — The Inference Stack

FlashAttention (recap for serving context)

- v1 (Dao et al., 2022): 2-4× speedup, O(N) memory — tile computation into GPU SRAM
- v2 (Dao, 2023, arXiv:2307.08691): 50-73% theoretical max FLOPS on A100
- v3 (Shah et al., 2024): H100 features — FP8, TMA, warp specialisation
- **Now default everywhere.** If you're not using FlashAttention in 2025, you're wasting money.

PagedAttention / vLLM (Kwon et al., 2023, arXiv:2309.06180)

- **Problem:** KV cache is variable-length and wasteful — pre-allocating for max length wastes ~60-80% of GPU memory
- **Solution:** Manage KV cache like **virtual memory pages** — allocated on demand, freed when done
- **Result:** Near-zero memory waste → 2-4× more concurrent requests per GPU

Continuous Batching: Dynamically add/remove requests mid-generation. No waiting for the longest sequence.

> **The 2025 minimum viable serving stack:** *vLLM/TGI + FlashAttention + PagedAttention + continuous batching + (optionally) speculative decoding.*

Distillation — How Far Can We Go?

✅ Available Now (deployed in production)

- Reasoning distillation via CoT traces (DeepSeek-R1 recipe)
- Synthetic data pipelines from frontier models (Phi-3, Orca, Gemma)
- Progressive distillation across model size ladder
- Quantization + distillation composition

🔶 Active Research (early results, promising)

- Multi-teacher distillation (ensemble of diverse teachers → single student)
- Distillation-aware training (design the teacher to be a better teacher)
- Process reward models for self-improving distillation loops
- Cross-architecture distillation (Transformer teacher → SSM student)

🔵 Blue Skies (fundamental, high potential)

- **Information-theoretic limits:** What is the minimum model size to capture a given capability?
- **Online distillation:** Student improves continuously as teacher improves
- Lottery ticket identification → targeted extraction without training the full teacher
- **Self-distillation:** A model improving by learning from its own best outputs (connects to RL)

3

Smarter, Not Bigger

Local Agents & Intelligence Per Watt

"We are in the era of just make it bigger. That era will end — not because scaling fails, but because it is not how intelligence actually works."

The Scaling Monoculture — The Problem

Since 2020, the dominant recipe: More parameters → better. More data → better. More compute → better.

This works. And that is precisely the problem.

- Funding flows to scale, not novelty
- Researchers optimise for GPT-N+1 instead of questioning the architecture
- Hardware roadmaps ossify around dense matrix multiply
- **Academic labs cannot compete** → the field becomes industrially captured

What the scaling laws actually say: They describe *one architecture family* on *one data type*. They say this is predictable, not optimal.

Signs of Diminishing Returns (2024-2025)

- GPT-4 → GPT-4o: modest improvements despite enormous investment
- DeepSeek-V3 achieved GPT-4-class with **~$6M training** by being smarter, not bigger
- Biggest gains from better data, better algorithms — not bigger models

> **The marathon analogy:** *Runner A consumes 10,000 cal/day but trains sloppily. Runner B consumes 2,000 cal/day but trains perfectly. Runner B wins — not more resources, but* **intelligently used** *resources.*

Intelligence Per Watt — The Right Metric

Current ML metrics (MMLU, Arena Elo) all measure capability-at-any-cost. **What if we measured differently?**

System	Power	Capability	Intelligence/Watt
🧠 Human brain	**~20W**	General intelligence, lifelong learning	**Absurdly high**
🖥️ GPT-4 inference (per query)	~3-10 kW·s	Broad knowledge, strong reasoning	Low
💻 **Llama 3.2-3B on phone**	**~2-5W**	Useful for many tasks	**Much higher**
📱 Specialised edge model	<1W	Expert at one domain	**Highest per niche**

- **2 billion smartphones** can run 3B models today — more aggregate compute than any cloud
- Data centre energy is a geopolitical and environmental issue — Microsoft buying nuclear reactors for AI
- Many applications need AI where there's no cloud: rural clinics, submarines, developing countries

> **The brain's trick:** *Compresses experience into reusable representations, maintains persistent state, builds hierarchical models — all on 20W, 3 bananas/day.*

What Small Models Can Do NOW (2025)

The floor of small model capability is rising faster than the ceiling of large models:

Model	Params	Teacher	Remarkable Achievement
Phi-3-mini	3.8B	GPT-4 (synthetic)	Rivals Mixtral-8×7B (46.7B total) on reasoning. Runs on a phone.
Gemma 2	2B / 9B	Gemini	2B competitive with models 5× larger
Llama 3.2	1B / 3B	Llama 3.1 405B	Strong instruction following, tool use, multilingual. On-device.
R1-Distill-Qwen-7B	7B	DeepSeek-R1 (671B)	55.5% AIME 2024 — competition maths. On a laptop.
Qwen 2.5-Coder-1.5B	1.5B	Larger Qwen	Competitive with CodeLlama-7B. Runs on a Raspberry Pi.

> 🔑 *The frontier model's purpose is shifting. It's not the product. It's the* **teacher**. *Its job is to discover intelligence; distillation's job is to deliver it to every device on earth.*

Local Agent Architecture

A personal AI agent running entirely on your hardware:

Component	Role	Example
Reasoning Core	Quantized LLM (3-7B, GGUF Q4)	Llama 3.2 3B, Phi-3-mini
Knowledge Retrieval	Embedding model + vector store	nomic-embed + ChromaDB/SQLite-VSS
Tool Use	File system, browser, APIs, code	Function calling
Memory	Persistent history, user preferences	Local SQLite
Orchestration	Planning, context management	Custom / LangChain

Runtime: llama.cpp / MLX / ExecuTorch · **Power:** 5-15W · **Latency:** <100ms/token · **Cost:** $0/query · **Privacy:** 100% local

> *This is not "ChatGPT on your laptop." This is an AI that knows your documents, learns your patterns, and never phones home.*

Why Local Wins — Five Pillars

🔒 **Privacy** — Data never leaves your device. Period. Medical records, financials, personal conversations — never sent to a third party. GDPR compliance is trivial.

⚡ **Latency** — No network round trip. 10ms local vs 500ms+ cloud. Critical for interactive use (coding assistants, autocomplete), robotics, real-time control.

💰 **Cost** — Zero marginal cost per query. A $500 device running 24/7 costs less than moderate API bills. Scales to billions of users without billions in infrastructure.

🌐 **Offline** — Planes, rural clinics, submarines, developing countries, bad WiFi. Resilient to cloud outages, API deprecation, provider pricing changes. **Sovereign** — no company can revoke your access.

🎯 **Personalisation** — Fine-tune on your data. Adapt to your style. Learn your domain. Compound returns: your agent improves the longer you use it.

> *"The best AI is the one that's always there, always private, and always yours."*

The Path Forward — Research Agenda

① Better Learning Signals

- Process reward models → dense, step-level supervision
- Self-play and verification loops → models that improve by checking their own work
- Curriculum learning → teach efficiently, like a tutor, not a fire hose

② Architecture Innovation

- SSMs (Mamba) → linear-time sequence modelling, natural for streaming
- MoE → activate only relevant parameters (DeepSeek-V3: 671B total, 37B active)
- Neuromorphic / event-driven computation → process only when something changes

③ Distillation as the Bridge

- Frontier models are **teachers, not products** — their purpose is to bootstrap smaller models
- DeepSeek-R1 proved reasoning is distillable; what about planning? Creativity? Common sense?

④ Efficiency Through Fundamental Insight

- BitNet b1.58: we may not need floating point — just {-1, 0, 1}
- Speculative decoding: 3× free speedup with the right small model
- Sparsity: the brain is >99% sparse at any moment

Open Research Questions

Fundamental 🧠

- What are the **information-theoretic limits** of distillation? Minimum model size for a given capability?
- Which capabilities compress cheaply (factual recall) vs expensively (reasoning, planning)?
- Are there **phase transitions** in model size below which capabilities vanish?

Architectural 🏗️

- Can SSMs match Transformers at 1/10th the compute for general language?
- Modular models: plug in domain-specific modules without retraining?
- Efficient continual learning without catastrophic forgetting?

Systems 📱

- On-device fine-tuning with limited memory?
- Federated learning for local agents to improve collectively without sharing private data?
- Hardware co-design for ternary/sparse computation? (BitNet implies radically different chips)

Evaluation 📊

- Benchmarks for efficiency: not "score at any cost" but "score at 5W / at 1B params / at $0 API"
- How to evaluate personalisation quality?

Call to Action — Download Ollama Tonight

After this lecture, do **one thing**:

```
# Install Ollama (macOS/Linux — one command)
curl -fsSL https://ollama.ai/install.sh | sh

# Run a 7B reasoning model on your laptop
ollama run deepseek-r1:7b

# Ask it to solve a maths problem. Watch it think.
```

Then reflect:

- This model was **distilled from a 671B-parameter system**
- It's running on your hardware, with no internet, no API key, no cost per query
- Two years ago, this capability didn't exist at any price
- **The gap between frontier and local is closing faster than anyone predicted**

> *The best time to start building local AI was two years ago. The second best time is* **today**.

Key Tools & Frameworks

Training & Fine-Tuning

- **axolotl** — github.com/axolotl-ai-cloud/axolotl — fine-tuning & distillation
- **Unsloth** — github.com/unslothai/unsloth — 2× faster, 60% less memory
- **HuggingFace TRL** — RLHF, DPO, distillation pipelines
- **vLLM** — github.com/vllm-project/vllm — production LLM serving

Local AI Tools

- **Ollama** — ollama.ai — `ollama run llama3.2`
- **LM Studio** — lmstudio.ai — GUI for model exploration
- **llama.cpp** — github.com/ggml-org/llama.cpp — local inference, GGUF
- **Open WebUI** — github.com/open-webui/open-webui — ChatGPT-like local UI

Frontier Model Demos

- ChatGPT (chatgpt.com) · Claude (claude.ai) · Gemini (gemini.google.com)
- **Groq** (groq.com) — custom LPU hardware, extremely fast inference
- **DeepSeek** (chat.deepseek.com) — open-weight reasoning models
- **Perplexity** (perplexity.ai) — search + LLM synthesis

HuggingFace

huggingface.co — the arXiv of models. Explore Spaces for demos, Models for weights, Datasets for training.

Key Papers — Your Reading List

Architecture & Training

- "Attention Is All You Need" — Vaswani et al. (2017) — arXiv:1706.03762
- "Training Compute-Optimal LLMs" (Chinchilla) — Hoffmann et al. (2022) — arXiv:2203.15556
- "RoFormer: Rotary Position Embedding" — Su et al. (2021) — arXiv:2104.09864
- "GLU Variants Improve Transformer" — Shazeer (2020) — arXiv:2002.05202

Alignment

- "Direct Preference Optimization" — Rafailov et al. (2023) — arXiv:2305.18290
- "Training LMs to Follow Instructions" (InstructGPT) — Ouyang et al. (2022)

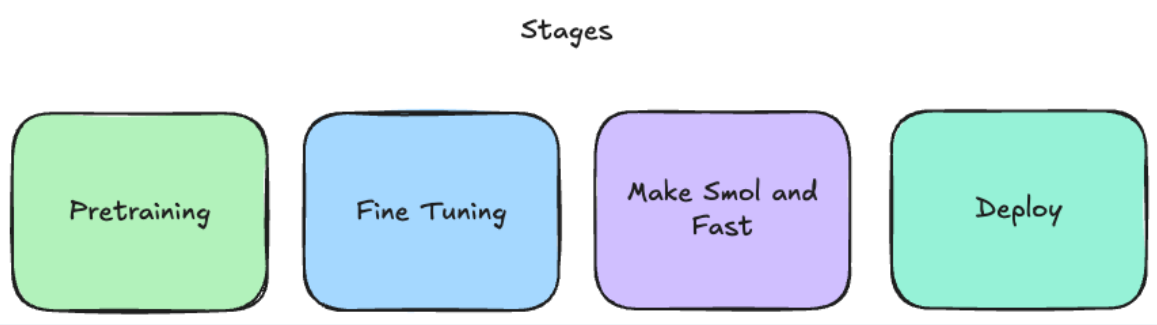
Efficiency & Compression

- "FlashAttention-2" — Dao (2023) — arXiv:2307.08691
- "GPTQ" — Frantar et al. (2022) — arXiv:2210.17323
- "AWQ" — Lin et al. (2024) — arXiv:2306.00978
- "BitNet b1.58" — Ma et al. (2024) — arXiv:2402.17764
- "SparseGPT" — Frantar & Alistarh (2023) — arXiv:2301.00774

Distillation & Small Models

- "Distilling the Knowledge in a Neural Network" — Hinton et al. (2015) — arXiv:1503.02531

Big Picture: High Level Stages



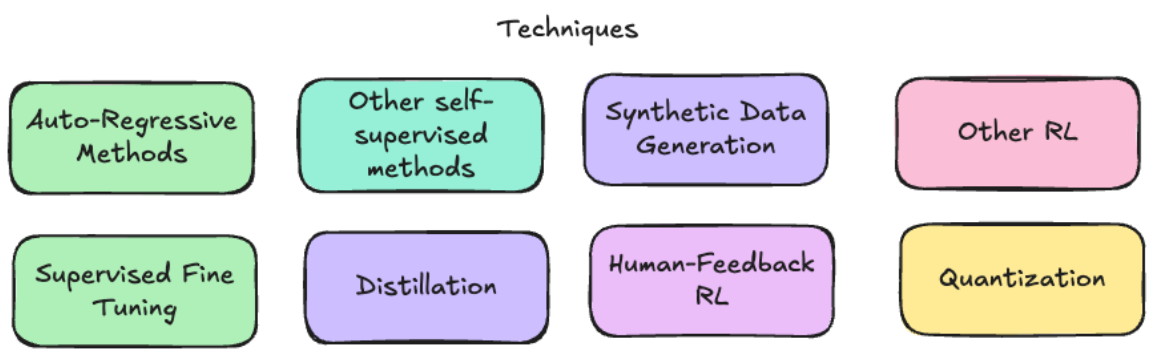
This diagram from Antreas's original lecture presents the **high-level pipeline stages** that connect the concepts we've covered:

- From **data collection and tokenisation** through pretraining, fine-tuning, and alignment
- To **compression and distillation** for efficient deployment
- Finally to **local / edge deployment** and personalisation loops

Each stage offers its own efficiency levers — the techniques we've explored apply at different points in this pipeline.

> *The colour-coding in this diagram represents the degree of relatedness between stages — not the section colours used in this slide deck.*

Big Picture: Underlying Technique Components



This diagram maps the **underlying technique components** that power each stage of the pipeline:

- Each box represents a family of techniques (attention, normalisation, training objectives, compression methods...)
- Colours indicate relatedness of one technique cluster to another
- Searching for any of these technique names + "LLM" will surface the key papers

Practical guidance: When studying any of these methods, replace "other" with "LLM" in your search query — this surfaces the LLM-specific implementations and benchmarks.

> *This overview diagram is your* **map of the field**. *You now know enough to navigate it independently.*

The Story We Told Today

Section 1 — LLM 101: The Transformer: LEGO bricks stacked into a sandwich, trained by predicting the next word trillions of times. The 2025 recipe: RMSNorm + SwiGLU + RoPE + GQA. Deceptively simple. Wildly powerful. Plus: how to *train* the beast at scale (FSDP) and fine-tune it efficiently (LoRA, QLoRA, DPO).

Section 2 — Compression & Distillation: The scaling monoculture is unsustainable. But we have tools: pruning (carve the statue), quantization (approximate π), distillation (teach the student). Combined: 100×+ compression. DeepSeek-R1 proved even *reasoning* is distillable.

Section 3 — Smarter, Not Bigger: Small models trained with distillation now rival giants from 12 months ago. The future is local, private, efficient. Intelligence per watt, not intelligence per dollar.

Section 4 — Resources: Tools, papers, and the big picture diagrams to navigate the field.

> **The brain runs on 3 bananas. AI runs on 1,500. Closing that gap is the most important research challenge of our generation.**

Thank You & Contact

Antreas Antoniou

- 📧 Email: [provided during lecture]
- 🐦 Twitter/X: @AntreasAntoniou
- 🏢 Axiotic AI — open-science research
- 🎓 University of Edinburgh — School of Informatics

Questions? Catch me after the lecture. Distillation theory, local agent systems, meta-learning connections, AI research careers — I'm here for all of it.

"The future of AI is not bigger models in bigger data centres. It's smarter models on every device."

Remember: Download Ollama tonight. The revolution fits on your laptop.