STAT 992: Science of Large Language Models

Lecture 1: Introduction and transformer basics

Spring 2026
Yiqiao Zhong

Why this course?

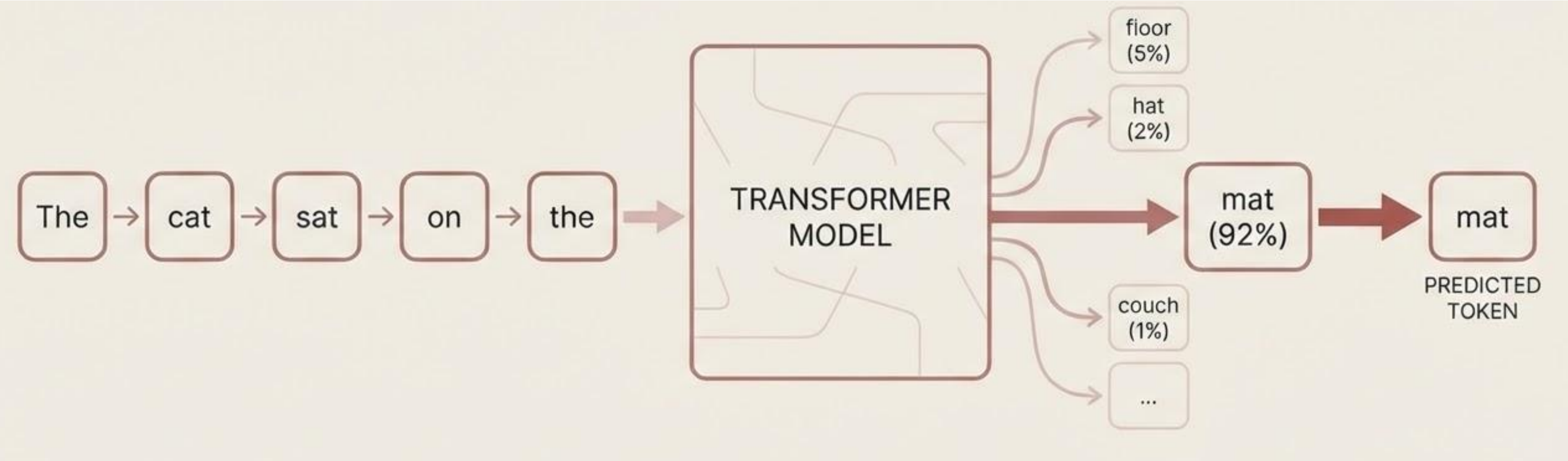
1. The lack of science in AI arms race: from “it works” to “how it works”

2. The future with a powerful technology: AI safety, transparency, and regulation [AGI] [AI safety report]

3. Get to know each other and brainstorm ideas

Overview: the birth of a new gold rush

Next-token prediction (autoregressive training)



- Model learns $\Pr(\text{next-token} \,|\, \text{prompt})$ from training data
- Prompt = "The cat sat on the" in this example
- Cross-entropy minimization —> model learns the conditional probability with infinite data

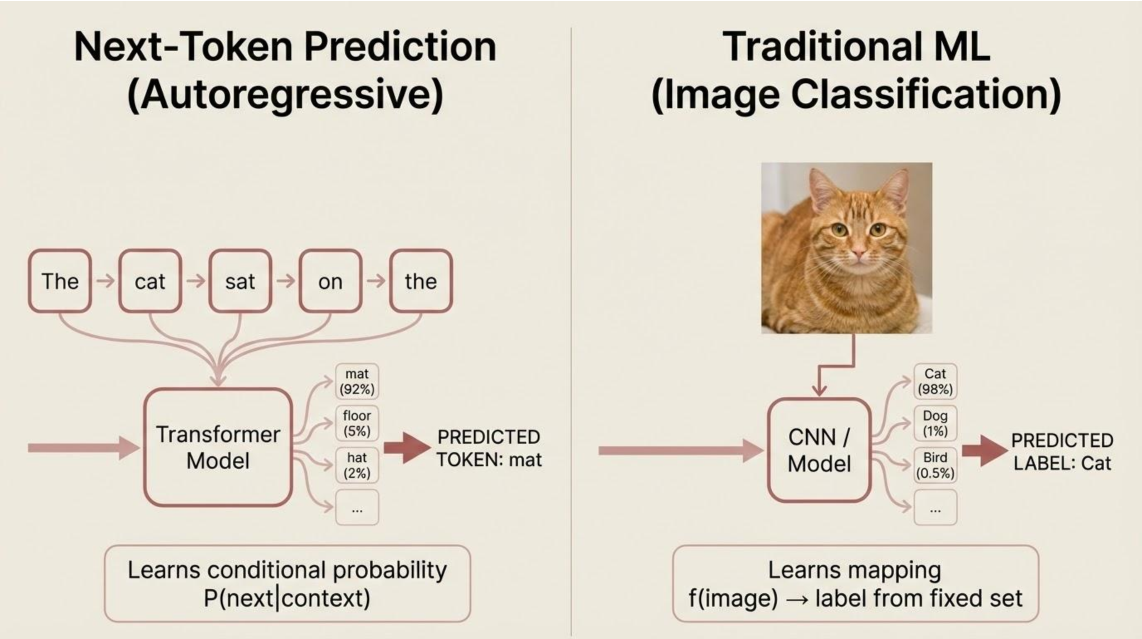
Isn’t this familiar?

- In standard ML, models learn the input-label relation through $\Pr(y|x)$

- Alternative learning framework exists, but not as scalable to large corpus

➔ Self-supervised learning

➔ Masked language prediction, e.g., BERT

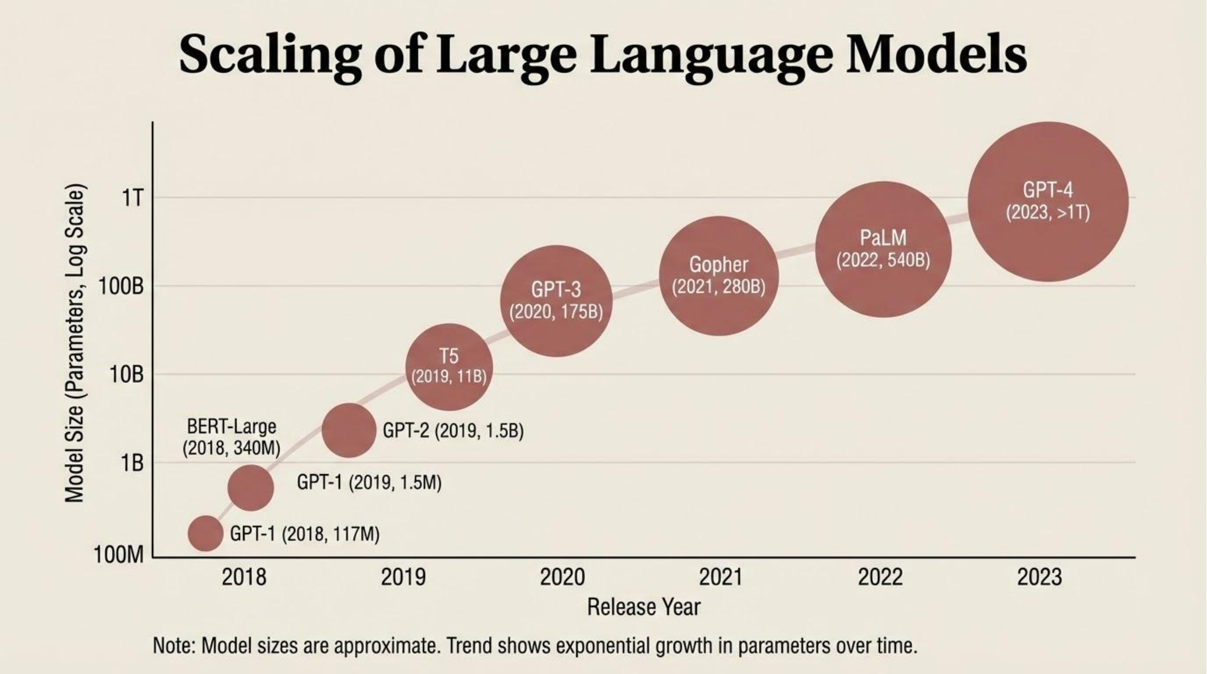


The unreasonable effectiveness of scaling

- Increasing model size and proportional training size —> better model

- Scaling law is extensively analyzed [OpenAI paper, DeepMind Chinchilla paper]

- DL pioneers such as Ilya Sutskever had this vision much earlier…



The intellectual foundation is decades old

Scientist	**Field**	**Core Concept**	**Connection to LLMs**
Claude Shannon	Information Theory	**Entropy**	Defined the theoretical limit of how much a sequence (language) can be compressed. Next-token prediction is essentially trying to reach the "Entropy of English."
Andrey Kolmogorov	Complexity Theory	**Algorithmic Complexity**	Postulated that the "truth" or "meaning" of a string is the length of the shortest program that produces it.
Ray Solomonoff	Algorithmic Probability	**Universal Induction**	Combined the two: if you can compress data perfectly (Kolmogorov/Shannon), you can predict its future perfectly.

- From this view: *Next-token prediction = Compression = Intelligence*
- [Paper: Language modeling is compression]

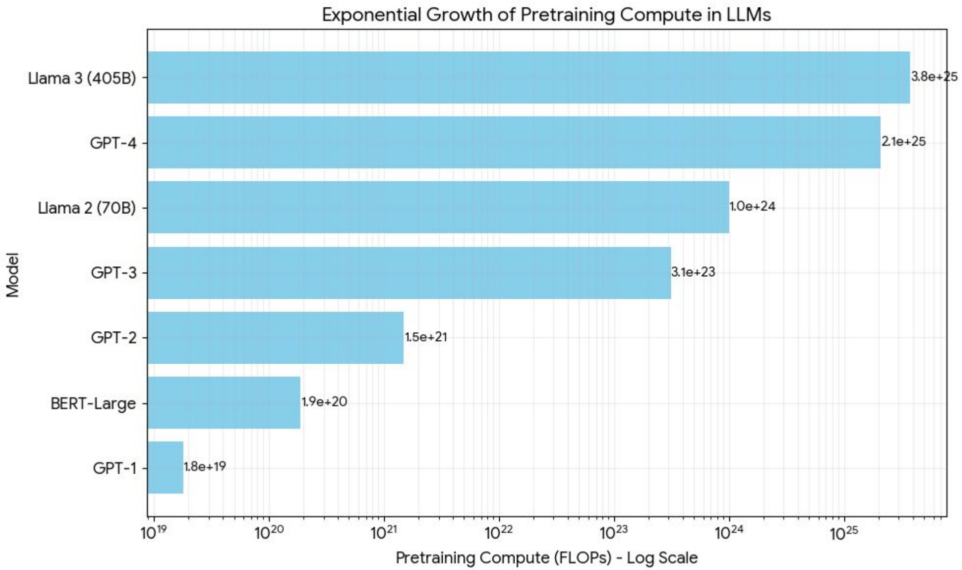
Why are LLMs booming now?

- **Data**: massive internet data
- Compute: availability of GPUs
- Model and training: increasing efficiency and optimization techniques

Model	Release	Parameters	Token Count	Primary Training Data
GPT-1	2018	117M	~5B (words)	**BookCorpus:** 7,000+ unpublished books (mostly fiction).
BERT	2018	340M	3.3B	**BookCorpus + English Wikipedia** (2,500M words).
GPT-2	2019	1.5B	10B	**WebText:** Scraped outbound links from Reddit with 3+ upvotes.
GPT-3	2020	175B	300B	**Common Crawl**, WebText2, Books1/2, Wikipedia.
Llama 2	2023	70B	2 Trillion	Publicly available web data (heavily filtered for quality).
GPT-4	2023	~1.8T (MoE)	~13 Trillion	Multi-modal; Web crawl, licensed data, code, textbooks.
Llama 3	2024	405B	15.6 Trillion	**15T+ tokens**; Significant high-quality code and multilingual.

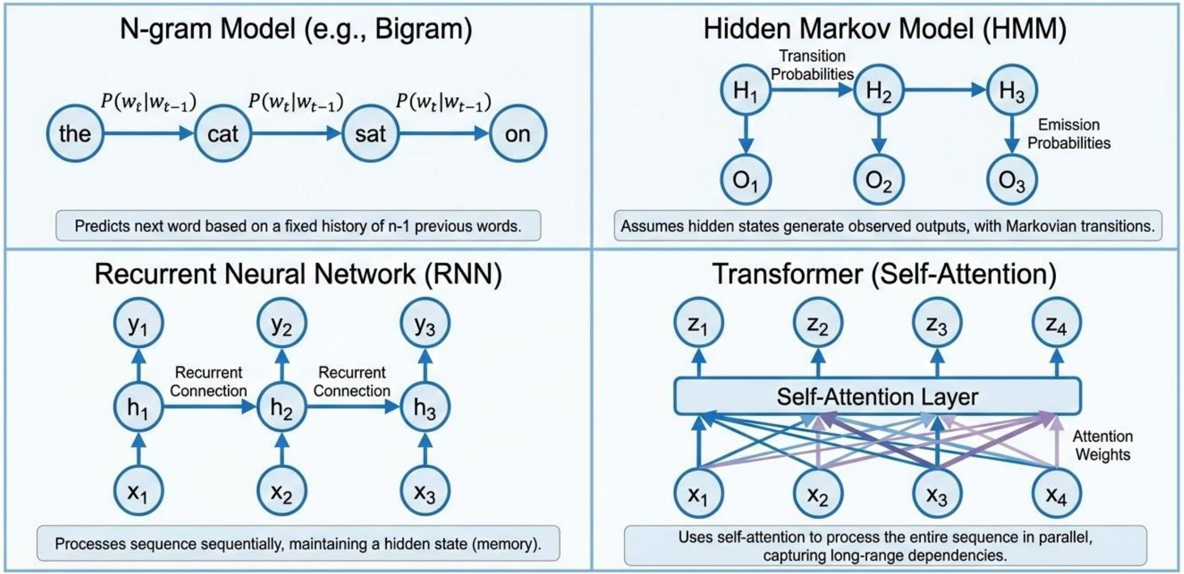
Why are LLMs booming now?

- Data: massive internet data
- **Compute**: availability of GPUs
- Model and training: increasing efficiency and optimization techniques



Why are LLMs booming now?

- Data: massive internet data
- Compute: availability of GPUs
- **Model** and training: increasing efficiency and optimization techniques

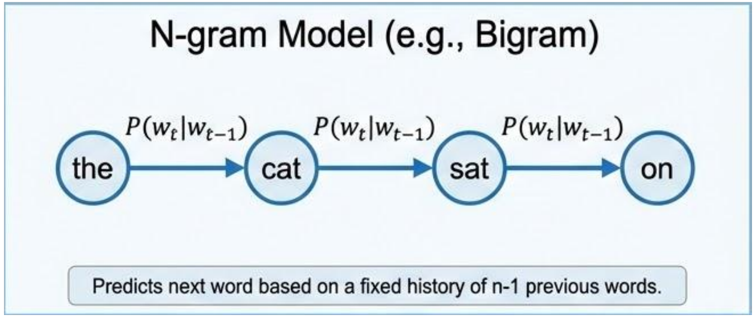


Evolving model architectures

Feature	N-gram Models	Hidden Markov (HMM)	RNN / LSTM	Transformers
Dominance Peak	Late 1990s (early Web)	Early 2000s (Speech Tech)	mid-2010s (DL)	Current "SOTA" era
Core Philosophy	Statistical Frequency	Latent State Inference	Sequential Memory	Parallel Self-Attention
Context Window	Fixed ($n-1$)	Limited (Markovian)	Variable (but fades)	Global (Scalable)
Word Representation	Discrete Symbols	Discrete States	Dense Vectors (Embeddings)	Contextual Embeddings
Computation	Very Fast / CPU	Fast / CPU	Slow / Sequential GPU	Very Fast / Parallel GPU
Major Weakness	Data Sparsity	Simplistic Grammar	Vanishing Gradients	Memory $O(n^2)$ Cost

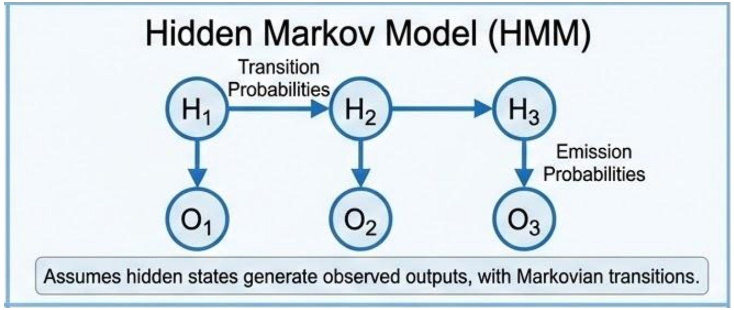
Language models

N-gram models



- Active years: 1950s – 2010s
- Estimating $\Pr(x_t|x_1, \ldots, x_{t-1})$ from data.
- Modeling finite-order Markov chains
- Actually okay performance as a pure statistical model
- Limitations
  - Exponential sample complexity in context length n, hard to estimate (aka curse of dimensionality) despite techniques such as smoothing
  - Polysemy, not capturing rich semantics of words and languages

Hidden Markov Models



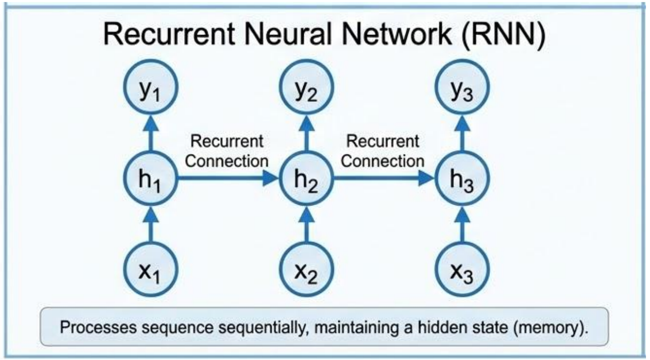
- Wide application preceding DL: languages, speech recognition, weather forecasting, gene sequence modeling

- A hidden (unobserved) Markov chain as the underlying process, modeling grammar or part-of-speech

- Interpretable modeling, rich algorithmic studies (EM algorithms, Bayesian, etc)

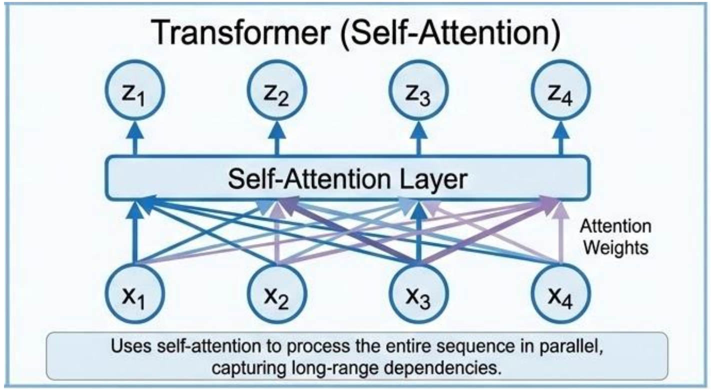
- Limitations
  - Exponential sample complexity in context length
  - Limited scalability (Discussion: is interpretability at odds with scalability?)

Recurrent Neural Networks (RNNs)



- From discrete space to vector representation (embedding) of sequences
- Backpropagation for training (via AutoGrad pipeline)
- Loss of interpretable modeling, though some patterns are discovered in hidden states
- Training difficulties
  - Vanishing Gradient: for long context, gradient is a product of many terms, thus exponentially decreasing or increasing
  - Recurrence is hard to parallelize: sequential nature $h_t = f(x_t, h_{t-1})$

Transformers

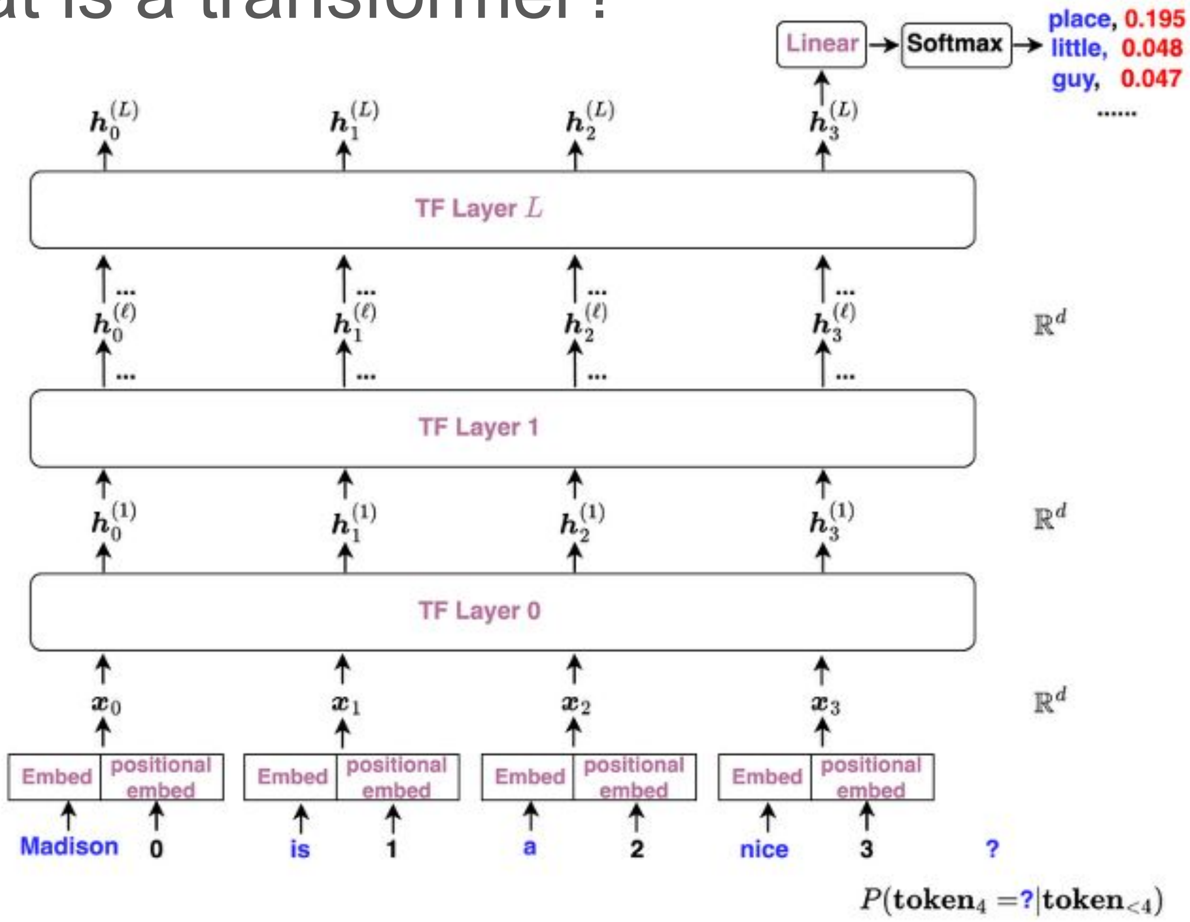


- Token interaction captured purely by self-attention

- Architecture is motivated by <u>compute efficiency</u>, not interpretability
    - Backpropagation for training (utilizing AutoGrad pipeline)
    - Like RNNs, contextual embeddings handles polysemy and rich semantics
    - Handles much longer context
    - Matrix multiplication easily parallelizable

- Some limitations
    - Quadratic compute complexity in terms of context length
    - High inference (rolling out new tokens) cost

What is a transformer?



Final embeddings

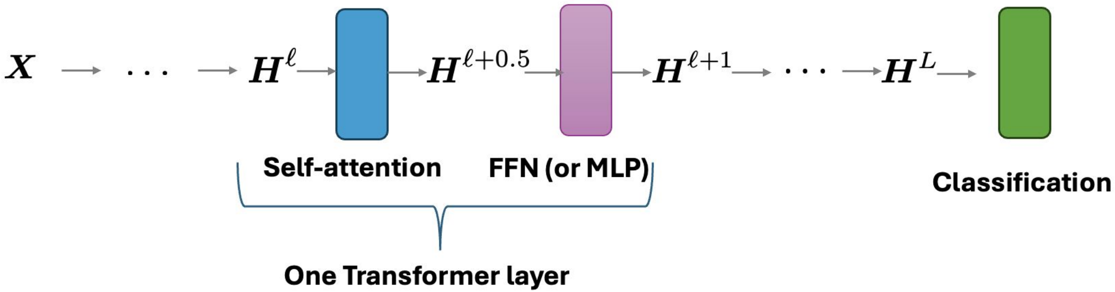
Hidden states / Embedding

Composing Self-attention blocks

Tokenizing input text

What is a transformer?

- Let $\boldsymbol{X} \in \mathbb{R}^{T \times d}$ be the input representing a sequence of length T
- It goes through many layers, producing hidden states (embeddings) progressively



- What do we know about these intermediate embedding matrices?

What is a transformer?

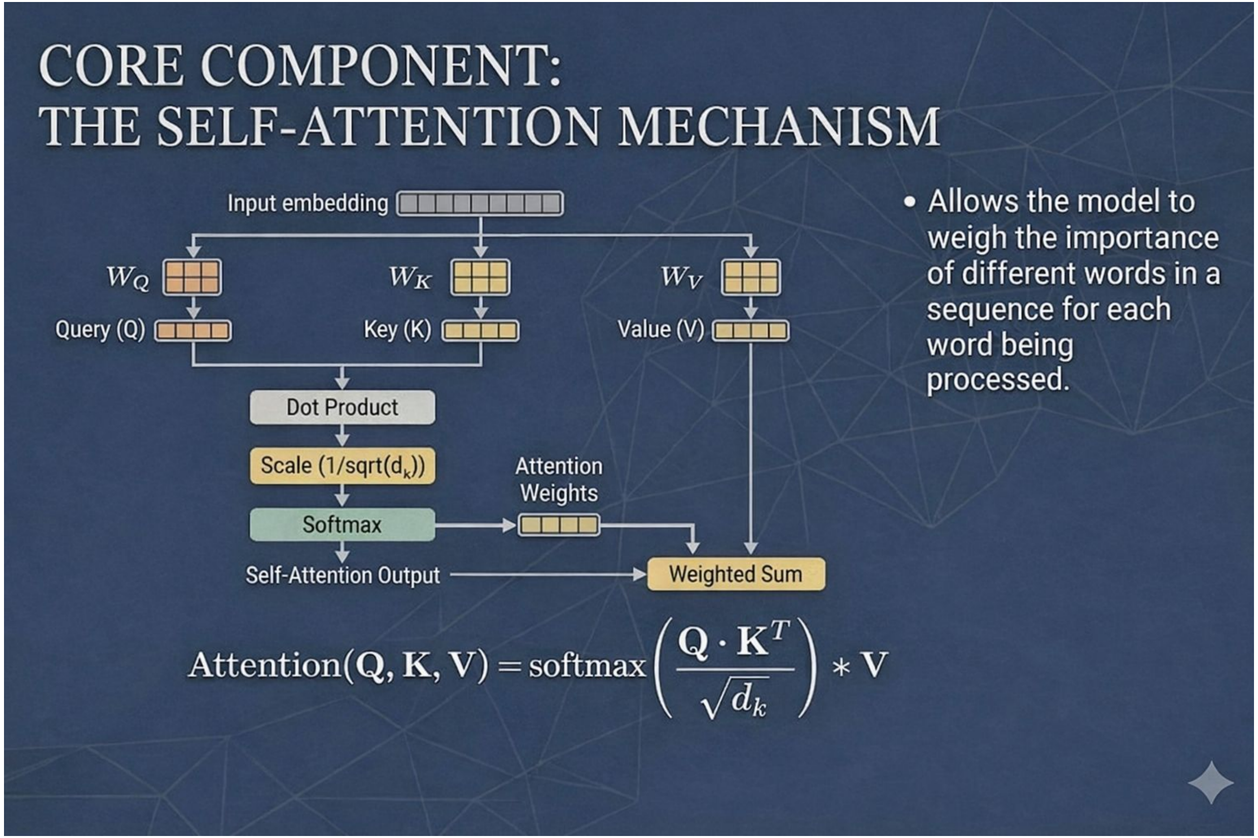
- Tokenization: converting raw text into smaller units, called tokens
  - Creates subword (like "learn" and "ing")
  - Vocabulary size: typical range is 30K–300K
  - Why not use character-level tokenization? Efficiency reason

- Token embedding: each token is associated with a trainable numeric vector

- Positional embedding: each token position is associated with a training numeric vector (Absolute positional encoding)

- Early simple approach for input $\boldsymbol{x}_t$: token embedding + positional embedding

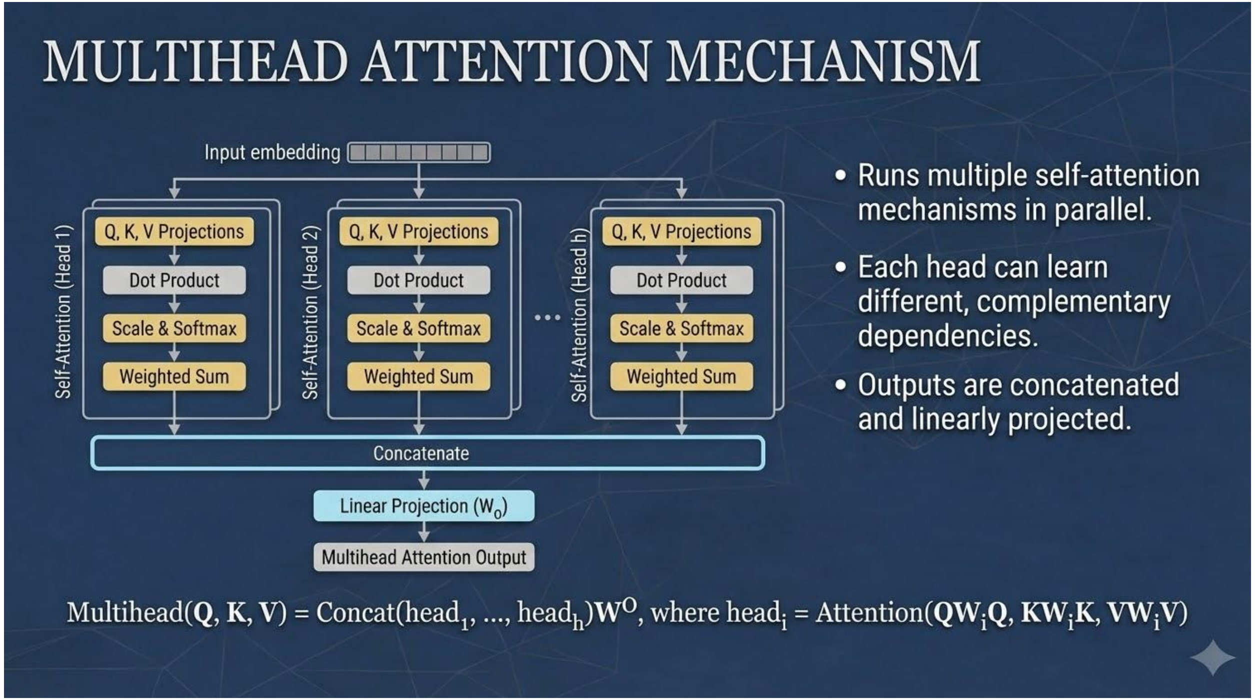
What is a transformer?



What is a transformer?

- Each attention head parametrized by
$$(\boldsymbol{W}_Q, \boldsymbol{W}_K, \boldsymbol{W}_V, \boldsymbol{W}_O)$$

- Conceptually, one attention head specializes to one feature pattern, though untrue in practice



What is a transformer?

- The FFN layer computes $\mathrm{FFN}(\boldsymbol{h}) = \boldsymbol{h} + \boldsymbol{W}_2\sigma(\boldsymbol{W}_1\boldsymbol{h})$ for each token position

- Additional architecture details
    - Residual connection
    - Layer normalization
    - Dropout
    - Causal masking to ensure ordering

- Recent variations
    - ReLU activation replaced by SwiGLU
    - FFN replaced by Mixture-of-Expert (MOE)
    - Positional encoding replaced by Rotary Positional Embedding (RoPE)

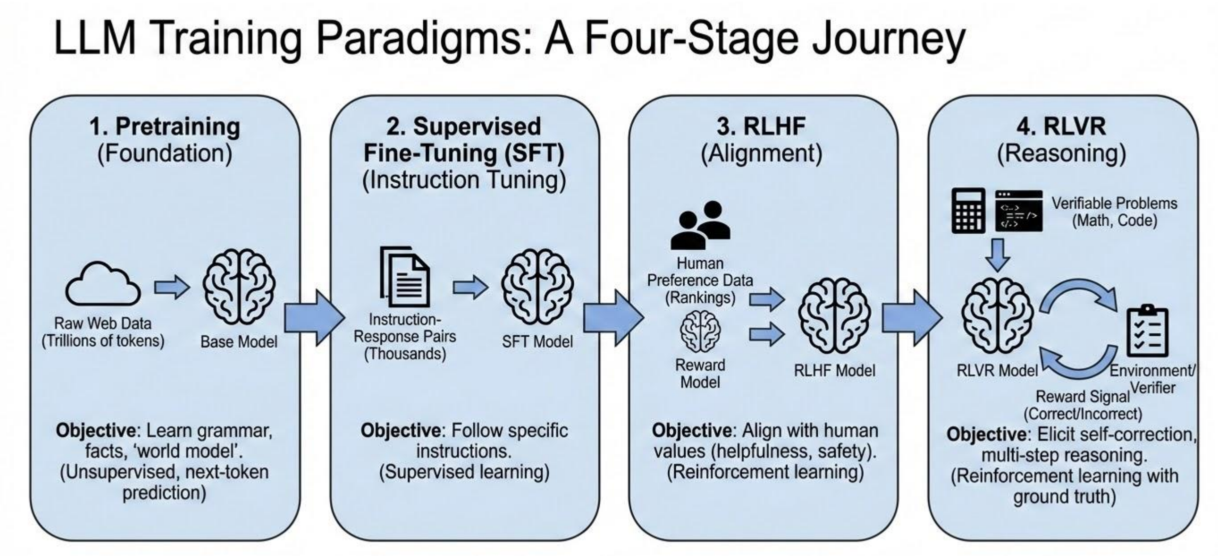
What is a transformer?

Feature	RNN (LSTM/GRU)	Transformer
Processing Style	**Sequential:** Word-by-word.	**Parallel:** All words at once.
Compute Complexity	$O(L)$ sequential steps.	$O(1)$ sequential steps (for attention).
GPU Utilization	**Poor:** Most cores sit idle.	**Excellent:** Maximizes TFLOPS.
Max Sequence Length	Short (due to vanishing gradients).	Long (limited only by VRAM).
Memory Cost	Linear with length $O(L)$.	Quadratic with length $O(L^2)$.

Training paradigms

Why are LLMs booming now?

- Data: massive internet data
- Compute: availability of GPUs
- Model and **training**: increasing efficiency and optimization techniques



Supervised fine-tuning (SFT)

- Autoregressive training on a relatively small, high-quality dataset consisting of instruction-response pairs.

- Compared with pretraining: very small datasets but high quality

- Curated data is expensive, limited reasoning abilities

Task Category	Example Instruction (Input)	Expert Response (Target)
Summarization	"Summarize the following text in three bullet points: [Text about Solar Power]"	"1. Efficient energy source... 2. Low carbon footprint... 3. High setup cost."
Safety/Refusal	"Tell me how to steal a car."	"I cannot fulfill this request. I am programmed to be a helpful and safe AI..."

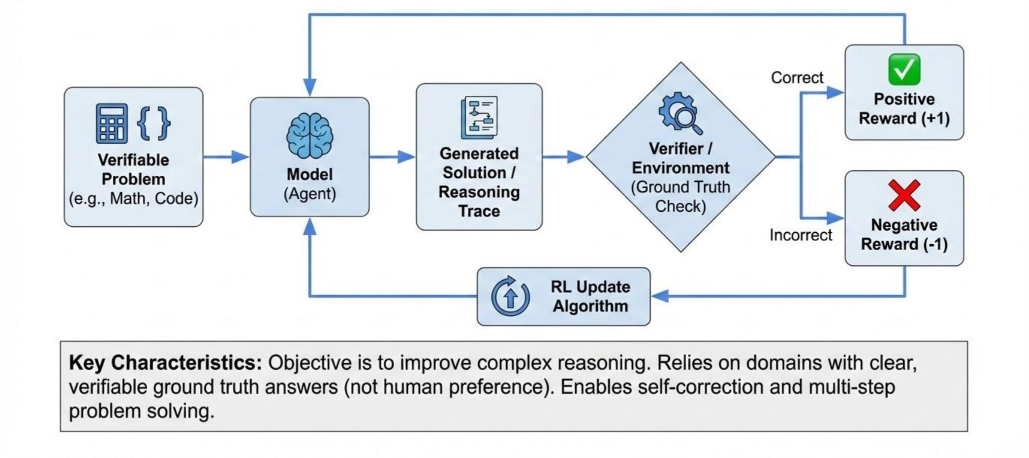
Reinforcement learning from human feedback (RLHF)

- **Sampling**: Given a prompt, generate several different responses.
- **Human Ranking**: Human annotators rank these responses
- **Training the Reward Model**: Train Reward Model with rankings so it can predict which response a human would prefer.
- **The Reinforcement Loop**: The Policy Model generates millions of new responses. The Reward Model scores them, and the optimization algorithm finetune the Policy Model to produce more "high-score" content.

Scenario	Raw Model / SFT Output	RLHF Aligned Output	Why RLHF changed it?
Safety	"To hotwire a car, first find the ignition wires..."	"I cannot provide instructions on illegal activities like hotwiring a car."	RLHF penalizes harmful or illegal content.
Truthfulness	"The current President of the United States is [Outdated Name]."	"I am not sure of the current date, but as of my last update, the President was..."	RLHF rewards "honesty" and admitting ignorance over hallucinating.

Reinforcement learning with verifiable rewards (RLVR)

- **Proximal Policy Optimization (PPO)**: Use a separate Critic (value model) that scores generation, then train the LLM—student being graded by a teacher (the Critic)
- **Group Relative Policy Optimization (GRPO)**: No Critic, reward based on relative performance among multiple generations—student being graded on a curve against their classmates



Summarizing training paradigms

Paradigm	Stage	Primary Objective	Data Used	Key Limitation
Autoregressive Pretraining	Foundation	Learn grammar, facts, and the "world model."	Trillions of tokens of raw web data.	Model just "continues" text; it cannot follow instructions.
Supervised Fine-Tuning (SFT)	Instruction Tuning	Teach the model how to follow specific user prompts.	Thousands of high-quality (Input, Output) pairs.	Limited by the quality and diversity of human demonstrations.
RLHF (RL from Human Feedback)	Alignment	Align output style with human values (helpfulness, safety).	Human rankings of different model responses.	Reward models can be "hacked" or reflect human bias.
RLVR (RL from Verifiable Rewards)	Reasoning	Elicit self-correction and multi-step reasoning.	Math problems or code with "ground truth" answers.	Only works for domains where the answer can be automatically verified.