An LLM doesn't know facts. It predicts the next token.

GenAI in 50 posts · Post 1/50

Swipe →

01

What an LLM is

A neural network trained to model patterns in language.

A causal LLM has one job: given the tokens so far, predict the next one.

EXAMPLE

"Large" means billions of learned parameters and huge training data. It does not mean the answers are checked facts.

02

It reads tokens, not words

A token can be a word, part of a word, punctuation or bytes. Cost and context limits count tokens.

EXAMPLE

"the" is usually 1 token. "unbelievable" might split into "un" + "believ" + "able". The split depends on the tokenizer.

03

The generation loop

- Score every token in the vocabulary
- Turn scores into probabilities
- Pick one token, append it
- Repeat until a stop condition

EXAMPLE

"The capital of France is" → " Paris" scores highest → appended → the next step predicts "."

04

Scores become probabilities

Raw scores are logits. Softmax turns them into probabilities that add up to 1.

EXAMPLE

Logits [2.0, 1.0, 0.1] → probabilities [0.66, 0.24, 0.10]. One point ahead in logits = 66% of the probability.

05

Softmax in 4 lines of Python

Exponentiate each logit, then divide by the total.

```
import math
logits = [2.0, 1.0, 0.1]
exps = [math.exp(x) for x in logits]
probs = [e / sum(exps) for e in exps]
# → [0.66, 0.24, 0.10]
```

06

Temperature reshapes the odds

Logits are divided by temperature T before softmax.

EXAMPLE

T = 0.5 → [0.86, 0.12, 0.02] sharper

T = 1.0 → [0.66, 0.24, 0.10]

T = 2.0 → [0.50, 0.30, 0.19] flatter

07

Temperature 0 is not truth

Low temperature makes output predictable. The most likely answer can still be wrong.

EXAMPLE

Ask about your company's internal leave policy. The model never saw it, so at T = 0 it gives the same confident guess every time.

08

Stage 1: pretraining

Learns broad patterns from huge datasets by making the correct next token more likely.

EXAMPLE

True next token got probability 0.25 → loss = −ln(0.25) ≈ 1.39.

At 0.9 → loss ≈ 0.11. Training pushes these up.

09

Stage 2: fine-tuning

Updates the weights for a narrower task. Preference training then favours answers people preferred.

EXAMPLE

Fine-tune on thousands of support Q&A pairs to learn the format. Raters then pick the more helpful of two answers.

10

Stage 3: inference

Fixed weights answer your request.
Using the model does not change it.

EXAMPLE

Chat with a model all day: nothing is learned permanently. A new session starts from the same weights.

11

Prompt examples are not training

Few-shot examples change this answer only. That's inference, not fine-tuning.

EXAMPLE

3 labelled tickets in the prompt improve classification now. Remove them and the improvement is gone.

12

Parameters vs context

- Parameters: what it learned in training
- Context: what you give it right now

EXAMPLE

Your refund policy changed last week? It isn't in the parameters. Put the current policy in the context.

13

Why LLMs hallucinate

- Missing knowledge (too new, niche or private)
- Missing or outdated context
- Pressure to answer instead of saying "I don't know"

14

Build around the model

- Measure on your real task
- Retrieve evidence, don't trust memory
- Keep permissions in code
- Plan for "I can't find that"

EXAMPLE

Policy assistant: the LLM writes, a retriever supplies current policy, code checks the user may see it.

15

Choosing a model

Skip the leaderboard. Compare candidates on your own cases: quality, latency, privacy, cost.

EXAMPLE

20 test questions, including ambiguous and unanswerable ones. Score correct answers, good refusals, p95 latency and cost per correct answer.

16

Interview questions

- Pretraining vs fine-tuning vs inference? Broad patterns vs narrower task vs fixed weights
- Does T = 0 stop hallucinations? No
- How to pick a model? Evaluate on your own cases

Full tutorial in the comments. Follow for Post 2/50: Transformers

Ali Hassan

linkedin.com/in/ali-hassan-here