UNDERSTANDING AI · WRITTEN EDITION · 2026-09-06
Day 3 · typeset 2026-09-10 22:53Z

One Token at a Time

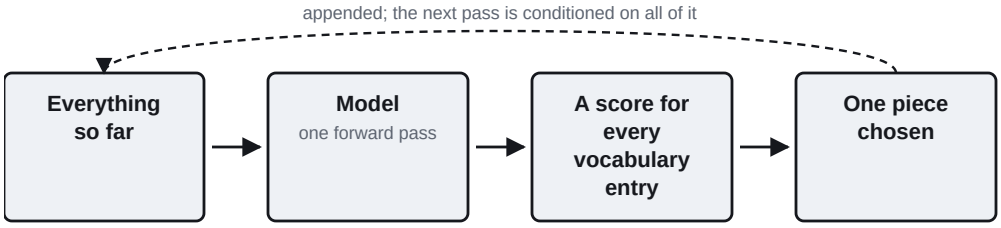
What "predict the next token" actually means — and the half of the sentence that almost every popular account leaves out.

Watch a modern language model answer a question and the words appear in pieces, left to right, at about the pace of somebody typing. That is more than a cosmetic touch, and less than a window onto the machine's thinking. Underneath, run the ordinary way, the system does build its answer one piece at a time, each piece selected with the earlier ones already fixed. That is a claim about the ordinary decoding loop rather than a law of the hardware; speculative decoding, exact and shipped, gets several accepted tokens out of a single large-model pass.

But the sentence has a second half, and it is the half that goes missing. The machine writes one piece at a time. It does not *read* one piece at a time: each choice is conditioned on everything already in front of it, and the positions that are already known are processed together in a single pass rather than one after another. (In ordinary generation the model does not recompute that prefix at every step — it reuses what it computed the first time.) Two measurements run on a single desktop machine on 6 September 2026 — described in full below; the scripts and outputs are kept with the episode's research record and are not published — put numbers on both halves, and the second number is the one that decides whether the comparison to autocomplete is fair.

The loop, and the one place it is forced

The loop, and the one place it is forced



Schematic. The forced step is the return arrow: the input to pass n+1 contains the output of pass n, so the passes cannot be reordered or run together. Everything to the left of it happens within a single pass, with every earlier position in view.

▼ Table view

The loop, and the one place it is forced — stages

#	Stage	Note
1	Everything so far	
2	Model	one forward pass
3	A score for every vocabulary entry	
4	One piece chosen	

The loop, and the one place it is forced — connections

From	To	Label
Everything so far	Model	
Model	A score for every vocabulary entry	
A score for every vocabulary entry	One piece chosen	
One piece chosen	Everything so far	appended; the next pass is conditioned on all of it

At each step the model receives everything it has been given so far and produces a score for every entry in its vocabulary — 50,257 entries for GPT-2, 248,320 for Qwen3.5-4B, the two models measured here. Those scores become probabilities. Something selects one. The selection is appended to the input, and the model runs again, conditioned on the longer text. The field's name for that loop is **autoregression**, a term the standard textbook flags as loose in its own footnote:

> Technically an autoregressive model predicts a value at time t based on a linear function of the values at times t − 1, t − 2, and so on. Although language models are not linear (since, as we will see, they have many layers of non-linearities), we loosely refer to this generation technique as autoregressive since the token generated at each time step is conditioned on the token selected by the network from the previous step.

(Jurafsky and Martin, Speech and Language Processing*, third-edition draft released 19 August 2026, chapter 7, footnote 4.)*

Only the return arrow in the diagram carries the dependency. Pass n+1's input contains pass n's output, so in the plain loop the passes cannot be reordered or merged; the dependency is on the chosen tokens, not on a fixed budget of model calls, which is the gap speculative decoding exploits. Everything to the left of that arrow happens within a single pass, with every earlier position in view.

Writing is sequential. Reading is not.

The distinction is easy to state and easy to check, so it was checked rather than asserted. Both models were loaded locally and instrumented with a counter on the top-level module, so the figures below are forward passes actually made rather than forward passes intended.

Reading and writing, counted on the same model in the same session

24
next-token distributions returned
from ONE forward pass over a 24-token sentence

1
forward pass used to produce them
logits tensor [1, 24, 50257]

20
forward passes to write 20 new tokens
exactly 1.00 per token

Counted with a hook on the top-level module, GPT-2 and Qwen3.5-4B, one desktop machine, 6 September 2026. Both models gave identical counts.

▼ **Table view**

Reading and writing, counted on the same model in the same session

Measure	Value
next-token distributions returned	24
forward pass used to produce them	1
forward passes to write 20 new tokens	20

A 24-token sentence handed to GPT-2 in one call returns a logits tensor of shape [1, 24, 50257]: a full next-token distribution at *every* position, produced by a single pass. Generating twenty new tokens from a prompt then costs exactly twenty passes, one per token. Qwen3.5-4B gave identical counts.

None of this is a new finding, and it is not offered as one. The same textbook states it plainly in its chapter on training:

> This means that all N positions in the context window can be scored at once against their true next tokens, giving N training examples from one pass through the network.

The measurement is here because the popular explanation of next-token prediction routinely implies that the model reads one piece at a time as well as writing one at a time, and that half is false. The asymmetry is also why these systems could be trained at all: had reading been as sequential as writing, a trillion tokens of training would have required a trillion sequential passes.

Two readings of the same fact, failing in opposite directions

The first is deflationary. Writing in *WIRED* on 29 December 2022, Gary Marcus put the popular version into one clause:

> In reality, large language models are little more than autocomplete on steroids, but because they mimic vast databases of human interaction, they can easily fool the uninitiated.

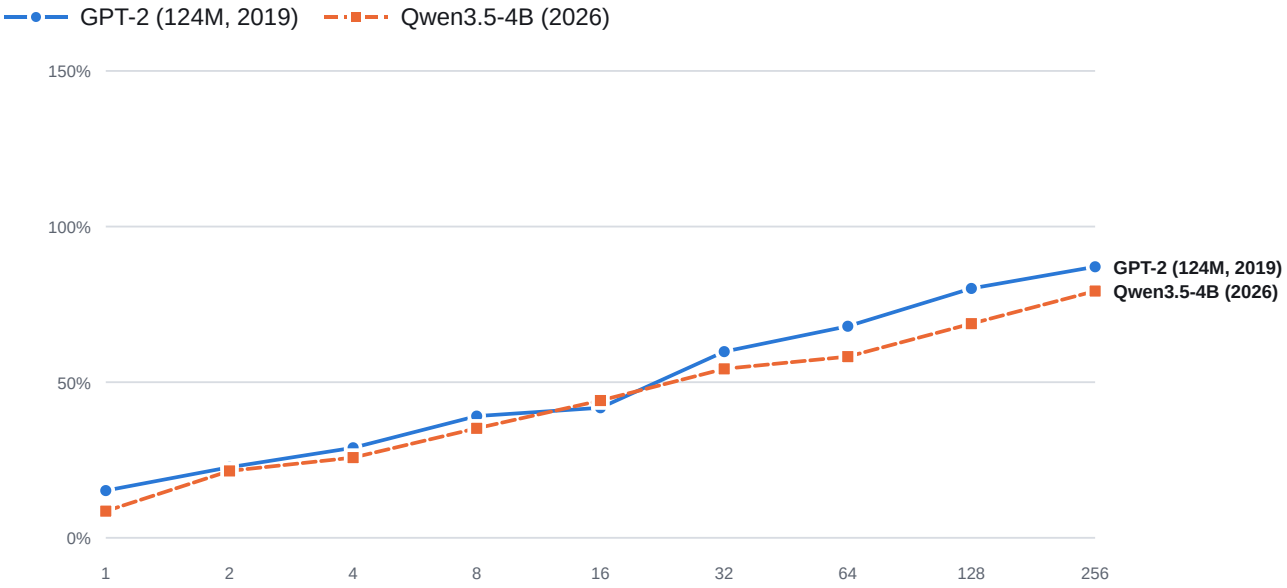
The mechanism in that description is not in dispute; the objective genuinely is next-token prediction. What is in dispute is the scope, and a paper by Zhicheng Lin published on 19 August 2026 locates the error precisely by granting the mechanism first:

> At the mechanistic level, current LLMs are conditional next-token predictors trained by cross-entropy minimization. This description, while accurate, becomes misleading when elevated into a complete account of what LLM-based systems are, or when used to dismiss them as cognitively trivial.

(Marcus's piece is about deception risk rather than decoder mechanics; the mechanistic reading is what the sentence has been taken to mean, not what its author set out to argue.)

The scope question has a number attached to it. Take the image the word autocomplete conjures — a prediction made from the previous word or two — and impose it on a real model. If the model is that thing scaled up, restricting it to that window should not change it much. (Real mobile keyboards are not that thing and have not been for years: Hard and colleagues described a recurrent neural next-word model for Google's keyboard in November 2018. The comparison below is to the folk image, not to a shipped product.)

How often the model's top choice matches the choice it makes with the whole passage



Same run, same 256 positions. At a window of two tokens — the window the folk image of autocomplete assumes — each model agrees with its own full-context choice about a fifth of the time. Truncating the input renumbers the surviving tokens from position zero, which is what a short prompt looks like to the model; holding the original positions instead puts the model far out of distribution and degrades it further (see the note on method).

▼ **Table view**

How often the model's top choice matches the choice it makes with the whole passage

	GPT-2 (124M, 2019)	Qwen3.5-4B (2026)
1	15.2%	8.6%
2	22.7%	21.5%
4	28.9%	25.8%
8	39.1%	35.2%
16	41.8%	44.1%
32	59.8%	54.3%
64	68%	58.2%
128	80.1%	68.8%
256	87.1%	79.3%

It changes it a great deal. Given two preceding tokens and nothing else, each model's top-scoring next token matches the token *it itself* selects with the whole passage in front of it about a fifth of the time: **22.7 per cent for GPT-2, 21.5 per cent for Qwen3.5-4B**. The same machine with a two-token window is, four times in five, a different predictor.

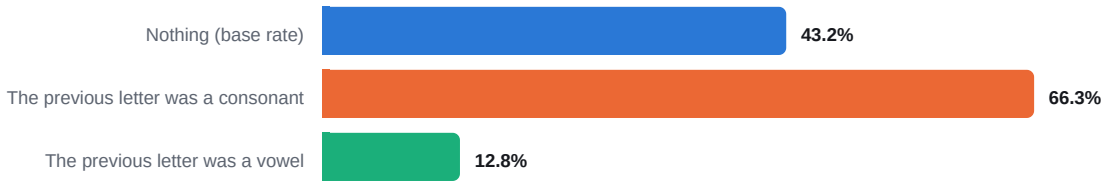
The second misreading runs the other way, and the moving cursor invites it: the words appear one at a time because the system is thinking them one at a time, in view. Three claims are routinely run together here and only the first is safe: that the next token has not yet been selected; that the model has computed nothing

about later words; and that it is doing nothing plan-like. Published evidence bears on the second and third and does not support them — Anthropic's causal-intervention study of 27 March 2025 reported that its model selects candidate rhyming words before writing the line that leads to them, and reported its own failure rate in the same passage, having *"found planned word features in about half of the poems we investigated"*. The display settles none of it: what it shows is a delivery setting, examined below, and an identical animation can sit on top of quite different schedules underneath.

Conditional probability, measured by hand in 1913

The idea underneath all of it is older than computers, and it was measured before there was anything to measure it with. On 23 January 1913 the Russian mathematician A. A. Markov lectured in St Petersburg on a count he had performed by hand: 20,000 letters from Pushkin's *Eugene Onegin* — the whole first chapter and sixteen stanzas of the second — each classified as a vowel or a consonant, then counted in pairs.

Markov's measurement: the chance the next letter is a vowel, Eugene Onegin, 20,000 letters



A. A. Markov, lecture of 23 January 1913; English translation in Science in Context 19(4), 591–600, December 2006. Letters classified vowel or consonant only, ъ and ь excluded; 8,638 vowels and 11,362 consonants. Markov's own figures are 0.432, 1104/8638 = 0.128 and 7534/11,361 = 0.663, and he writes the difference as δ = −0.535.

▼ **Table view**

Markov's measurement: the chance the next letter is a vowel, Eugene Onegin, 20,000 letters

Conditioned on	P(vowel)
Nothing (base rate)	43.2%
The previous letter was a consonant	66.3%
The previous letter was a vowel	12.8%

His own conclusion, in the published English translation:

> As we can see, the probability of a letter being a vowel changes considerably depending upon which letter – vowel or consonant – precedes it.

Two details are usually lost in retelling and both matter. First, Markov was not studying language and built no model of it: his paper is an argument about a *dispersion coefficient* — whether a chain of dependent trials spreads out the way the independent case predicts — and Pushkin supplied the test material, not the subject. He even flags his own terminology, noting that he deviates *"slightly from usual terminology, whereby we should have taken the square root of the number that we call the coefficient of dispersion"*.

Second, he did not stop at pairs. He counted vowel-vowel-vowel (115 cases) and consonant-consonant-consonant (505), producing second-order figures of 0.104 and 0.132 — two rungs of a ladder, by hand, thirty-five years before anyone climbed the rest of it.

The story most often attached to the paper — that Markov was refuting P. A. Nekrasov, who had argued that the law of large numbers requires independent trials and therefore that social statistics prove free will — does not appear in the 1913 text at all. It comes from Brian Hayes, *First Links in the Markov Chain*, *American Scientist* 101, March–April 2013, pages 92–97, and is a historian's reading of the episode rather than the mathematician's account of it.

The translation carries one further wrinkle worth a clause: the English text was rendered from a German intermediate. Its own footnote records that the paper was *"translated into German by Alexander Y. Nitussov, Lioudmila Voropai, and David Link; translated into English by Gloria Custance and David Link."*

Running the measurement forwards: Shannon, 1948

Markov measured a dependency. Claude Shannon ran a measurement like it *forwards*, and printed what came out. Section 3 of *A Mathematical Theory of Communication* — the July 1948 instalment, in *Bell System Technical Journal* volume 27 — is titled "The Series of Approximations to English", and it is a ladder.

Rung	What each symbol is conditioned on	Shannon's output (verbatim, in full)
1	nothing; 27 symbols, equiprobable	`XFOML RXKHRJFFJUJ ZLPWCFWKCYJ FFJEYVKCQSGHYD QPAAMKBZAACIBZLHJQD.`
2	letter frequencies	`OCRO HLI RGWR NMIELWIS EU LL NBNESEBYA TH EEI ALHENHTTPA OOBTTVA NAH BRL.`
3	the previous letter	`ON IE ANTSOUTINYS ARE T INCTORE ST BE S DEAMY ACHIN D ILONASIVE TUCOOWE AT TEASONARE FUSO TIZIN ANDY TOBE SEACE CTISBE.`
4	the previous two letters	`IN NO IST LAT WHEY CRATICT FROURE BIRS GROCID PONDENOME OF DEMONSTURES OF THE REPTAGIN IS REGOACTIONA OF CRE.`
5	word frequencies	`REPRESENTING AND SPEEDILY IS AN GOOD APT OR COME CAN DIFFERENT NATURAL HERE HE THE A IN CAME THE TO OF TO EXPERT GRAY COME TO FURNISHES THE LINE MESSAGE HAD BE THESE.`
6	the previous word	`THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH WRITER THAT THE CHARACTER OF THIS POINT IS THEREFORE ANOTHER METHOD FOR THE LETTERS THAT THE TIME OF WHO EVER TOLD THE PROBLEM FOR AN UNEXPECTED.`

The top rung is 35 words long, not the forty it is sometimes given as; the strings above are complete rather than trimmed, which is the only way the effect can be judged. Shannon's assessment of the climb:

> The resemblance to ordinary English text increases quite noticeably at each of the above steps.

Then comes the sentence that answers the harder question — how a machine choosing one piece at a time produces something that hangs together across a paragraph:

> Note that these samples have reasonably good structure out to about twice the range that is taken into account in their construction.

Coherence outruns the window. A two-word memory produced a ten-word run that Shannon judged *"not at all unreasonable"*. His conclusion keeps its hedge — *"It appears then that a sufficiently complex stochastic process will give a satisfactory representation of a discrete source"* — and nothing in the paper asserts that language *is* such a process.

The apparatus was a shelf and a pencil. Rungs 1 and 2 used a book of random numbers and a published frequency table; rungs 3 to 6 used a physical procedure Shannon describes exactly:

> To construct (3) for example, one opens a book at random and selects a letter at random on the page. This letter is recorded. The book is then opened to another page and one reads until this letter is encountered. The succeeding letter is then recorded.

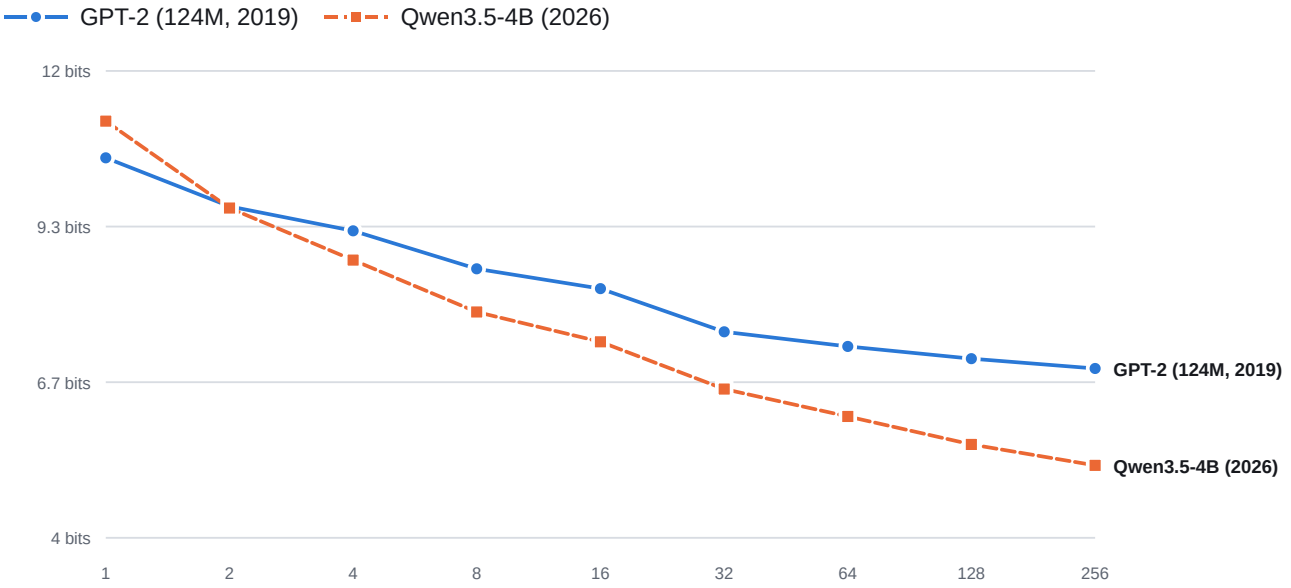
He stopped there, and said why:

> It would be interesting if further approximations could be constructed, but the labor involved becomes enormous at the next stage.

The ladder, climbed the rest of the way

The labour is no longer enormous. The same ladder was therefore run on 6 September 2026 against two published models — GPT-2 (124M parameters, 2019) and Qwen3.5-4B (4.2bn parameters, 2026) — over a 1,024-token passage, with 256 positions scored at every rung. The only variable is how many preceding tokens each model was allowed to see.

Shannon's ladder, run forward: bits per token against how much text the model may see



Measured on one desktop machine, 6 September 2026, over 256 scored positions of a passage of English prose written in August 2026 — after both models' training data closed. The x-axis is the number of preceding tokens the model was given. Bits are not comparable BETWEEN the two models, whose tokenisers cut the same text into different numbers of pieces; the comparison that holds is each curve against itself. Shannon reached the second rung by hand.

▼ Table view

Shannon's ladder, run forward: bits per token against how much text the model may see

	GPT-2 (124M, 2019)	Qwen3.5-4B (2026)
1	10.5 bits	11.1 bits
2	9.7 bits	9.7 bits
4	9.3 bits	8.8 bits
8	8.6 bits	7.9 bits
16	8.3 bits	7.4 bits
32	7.5 bits	6.5 bits
64	7.3 bits	6.1 bits
128	7.1 bits	5.6 bits
256	6.9 bits	5.2 bits

The curve falls at every rung, for both models, all the way to a window of 256 tokens. Shannon reached the second rung. There was a great deal further to go.

The full figures, in bits per token, top-1 accuracy against the true next token, and agreement with the model's own full-context choice:

Window	GPT-2 bits	GPT-2 acc.	GPT-2 agree	Qwen bits	Qwen acc.	Qwen agree
1	10.51	.070	.152	11.14	.039	.086

Window	GPT-2 bits	GPT-2 acc.	GPT-2 agree	Qwen bits	Qwen acc.	Qwen agree
2	9.68	.086	.227	9.65	.090	.215
4	9.26	.098	.289	8.76	.129	.258
8	8.61	.121	.391	7.87	.188	.352
16	8.27	.148	.418	7.36	.238	.441
32	7.53	.199	.598	6.55	.246	.543
64	7.28	.231	.680	6.08	.277	.582
128	7.07	.215	.801	5.60	.305	.688
256	6.90	.223	.871	5.24	.336	.793
whole passage	6.79	.234	1.000	5.26	.324	1.000

On method, because the numbers are only worth what the method is. The passage was written in August 2026 on the machine that ran the test, after both models' training data closed, so memorisation cannot be producing the effect; a second passage — Shannon's own 1948 prose, public for seventy-eight years and certainly inside both training sets — gives the same monotonic shape. Bits per token are not comparable *between* the two models, whose tokenisers cut identical text into different numbers of pieces, so Qwen's 5.24 is not "better than" GPT-2's 6.79; each curve is comparable only against itself. Every cell rests on 256 scored positions from one passage, which makes these estimates and not a benchmark. And truncating the input to a window renumbers the surviving tokens from position zero — which is exactly what a short prompt looks like to a model, and is therefore the well-posed version of the question. The alternative, passing the tokens their original absolute positions, was tried and is worse: GPT-2 handed a two-token fragment at positions 400–401 with nothing before it collapses to 21.7 bits per token, because it has never been trained on such a state.

Delivery is a switch, and the switch is old

Sequential generation makes streaming *possible*. It does not make it happen. OpenAI's own developer documentation states the default:

> By default, when you make a request to the OpenAI API, we generate the model's entire output before sending it back in a single HTTP response.

The switch predates the chatbot era by a wide margin. The first commit of OpenAI's Python client — `3c6d4cd6`, authored by Greg Brockman, 25 October 2020 — already carries server-sent-event parsing with the `data: [DONE]` sentinel still in use today, and a command-line flag whose help string reads *"Stream tokens as they're ready."* That is twenty-five months before ChatGPT. It is not a priority claim: a surviving dated artifact proves existence by that date and not invention on it, public forum threads discuss streaming responses as early as August 2021, and the archived API-reference pages survive only as empty JavaScript shells, so an earlier public artifact cannot be ruled out.

Once a process is shown to a user, it acquires a number: the gap between one word and the next. MLCommons, the multi-vendor consortium behind the MLPerf benchmarks, has now written that number into its rules three times, and published a different justification each time.

Round, and the workload it applies to	Date	Time to first token	Time per output token	The stated anchor
Inference v4.0, **Llama 2 70B server**	27 Mar 2024	≤ 2 s	**≤ 200 ms**	*"A TPOT of 200 ms translates to a maximum allowed generation latency that maps to ~240 words per minute … which is often cited as the average human reading speed."*
Inference v5.0, **Llama 2 70B interactive**	2 Apr 2025	≤ 450 ms	**≤ 40 ms**	analysis of *"industry research, user surveys, and performance data from leading platforms like ChatGPT and Perplexity AI in late 2024"*
Inference v6.0, **DeepSeek-R1 interactive**	24 Mar 2026	≤ 1.5 s	**≤ 15 ms**	latency-optimised reasoning, with a named speculative-decoding configuration mandated to reach it

Three different workloads under one benchmark suite, so the three rows are a sequence of judgements rather than one figure tightening; what they share is that the anchor moved off the reader in the second round. Even in 2024 MLCommons scoped its own yardstick, writing in the next sentence that *"other use cases have tighter latency constraints"* and naming code generation and agents — and its 240-words-per-minute conversion assumes roughly 1.25 tokens per word, which is a property of a tokeniser rather than of reading.

Two attempts to escape the sequence

The first accepts the sequence and attacks the waiting. In speculative decoding a small, cheap model guesses the next several tokens and the large model checks all of them in a single pass — which it can, because checking is the parallel half. Guesses that survive the check cost no further large-model pass; the rest are discarded. What raises it above a heuristic is that the *distribution* is preserved exactly: a modified rejection-sampling step corrects the draft, so the fast path samples from the same distribution as the slow one. That is a distributional guarantee rather than a promise that a given run returns the same string it would otherwise have returned.

The speedups, and who measured them:

Source	What was measured	Reported	Conditions
Leviathan, Kalman & Matias, arXiv 2211.17192v1, 30 Nov 2022	their own method	**2×–3×** overall; 3.4×/2.6× on translation, 3.1×/2.3× on summarisation	T5-XXL 11B with a T5-small 77M drafter, **batch size 1, a single TPU-v4**
Chen, Borgeaud, Irving, Lespiau, Sifre & Jumper, arXiv 2302.01318, 2 Feb 2023	their own method	**2×–2.5×**	Chinchilla 70B, distributed; guarantee stated as preserving the target distribution *"within hardware numerics"*

Source	What was measured	Reported	Conditions
Liu, Yu, Park, Stoica & Cheung, arXiv 2601.11580, 31 Dec 2025	**somebody else's** method, on production vLLM	**1.96× → 1.21×**	five variants, four models, batch size 1 rising to 128, NVIDIA H100s

The third row is the one that matters, because the first two are authors measuring their own work. The Berkeley group's stated objection is that prior evaluations *"test at batch size 1 — an unrealistic setting that inflates speedup numbers"*, and its own figures fall from 1.96× with a single user to 1.21× once 128 requests share the machine. The first paper is explicit about the trade too, noting that latency improves *"at the cost of an increased number of arithmetic operations"* and that the method *"is not helpful for configurations where additional computation resources are not available."*

The second attempt appears to refuse the sequence — generate a block of text at once and refine it over several passes, as image models do. Google released such a model on 10 June 2026. Its own model card describes what it actually does:

> Once a canvas is fully denoised, it is processed by the encoder and appended to the KV cache, after which the model generates the next canvas. This block-autoregressive approach facilitates text generation at higher speeds.

Beyond 256 tokens the model commits a finished block and starts the next one conditioned on everything committed so far. The loop did not disappear; its stride widened. The same launch post carries the vendor's recommendation against its own release —

> For applications that demand maximum quality, we recommend deploying standard Gemma 4.

— and its model card puts numbers behind that sentence: on 15 published benchmarks the autoregressive model wins 14, the single exception being Humanity's Last Exam without tools (11.0 per cent against 8.7). The same post also scopes the headline speed claim to *"local and low-concurrency"* use, warning that in high-QPS cloud serving parallel decoding *"offers diminishing returns and can result in higher serving costs."*

Two independent attacks on the sequential bottleneck, then, and both weaken as the batch grows. The speed claim itself remains unaudited: 88 days after release, Artificial Analysis records *"No API providers are currently available for DiffusionGemma 26B A4B"*, and lists its output speed as N/A, because nobody outside Google serves it.

What to watch

Whether the speculative-decoding rule acquires a second vendor. MLPerf Inference v6.0's DeepSeek-R1 Interactive scenario is the first benchmark rule anywhere to require speculative decoding, and MLCommons has more than 130 members. Its published results file for that scenario carries three entries from two organisations — NVIDIA and GigaComputing — every one of them on NVIDIA accelerators,

against 18 Server and 19 Offline results from eight submitters. The rule is multi-vendor; the first round's practice was not. A third organisation, or any result on non-NVIDIA silicon, in the next results file would close that gap, and the file is public.

Whether an emitted word can ever be recalled. On 3 September 2026 OpenAI shipped mid-turn steering, which lets a caller send further instructions into a response that is still being generated. The same documentation states the boundary:

> Steering does not rewrite output already sent to your application, undo earlier actions, or cancel tools that have already started.

That is the constraint under discussion here, written down as a product limitation by the company shipping the feature. If it is ever lifted, that sentence is where it changes.

The idea to keep

A language model writes one piece at a time; each piece is conditioned on everything before it.

The first half is forced, and it accounts for the moving cursor, the wait before the first word, and an entire branch of engineering devoted to concealing the delay. The second half is what separates the machine from the autocomplete of popular imagery, and the separation is measurable: restricted to two tokens of context, both models tested here disagree with their own full-context judgement about four times in five.

The shape is old. A mathematician counted vowels in Pushkin by hand in 1913; an engineer opened books at random in 1948 and stopped because the labour became enormous. What fills the shape in, and what it costs to run, have changed beyond recognition. The shape itself has barely moved — and the most prominent attempt to escape it, a diffusion model shipped this June, turns out to have reintroduced it one level up.

Both are free to read: Shannon's paper, primary evidence from 1948 and where section 3 is the ladder, and Jurafsky and Martin's *Speech and Language Processing*, a textbook and therefore a secondary account of everything except its authors' own opinions.

What is not claimed

The measurements above show one machine given less text, not a model *trained* on short contexts; they answer how much of a prediction depends on how far back it can see, and say nothing about how good a genuinely short-context model would be. Nothing here establishes that autoregression is ending: no shipped frontier model has dropped it, the current textbook calls causal models *"the most common language models used in the world today"*, and the newest published work in the area uses diffusion to sample faster *from* an autoregressive distribution rather than to replace one. And on whether these systems plan, the only thing asserted is what Anthropic measured, at the rate Anthropic reported it.

Sources

Source	Date	Note
A. A. Markov — *An Example of Statistical Investigation of the Text Eugene Onegin*	lecture 23 Jan 1913; translation Dec 2006	*Science in Context* 19(4), 591–600; doi 10.1017/S0269889706001074
Brian Hayes — *First Links in the Markov Chain*	Mar–Apr 2013	*American Scientist* 101, 92–97; the Nekrasov account
Claude Shannon — *A Mathematical Theory of Communication*	Jul & Oct 1948	*BSTJ* 27; §3 is the ladder, in the July instalment
Jurafsky & Martin — *Speech and Language Processing*, 3rd ed. draft	released 19 Aug 2026	ch. 7 footnote 4; ch. 7 on parallel training; ch. 1 on prevalence
Direct measurement, one desktop machine	6 Sep 2026	GPT-2 and Qwen3.5-4B run locally; scripts and JSON kept with the episode's research record, not published
Zhicheng Lin — *Six misconceptions about large language models*	19 Aug 2026	arXiv 2608.20421; an arXiv perspective, no journal on the record
Gary Marcus — *The Dark Risk of Large Language Models*	29 Dec 2022	WIRED
Anthropic (Lindsey et al.) — *On the Biology of a Large Language Model*	27 Mar 2025	Transformer Circuits; planning in poems, with its own failure rate
OpenAI — *Streaming API responses*; *Mid-turn steering*	read 6 Sep 2026; steering shipped 3 Sep 2026	`developers.openai.com`
OpenAI — `openai-python` initial commit `3c6d4cd6`, Greg Brockman	25 Oct 2020	SSE parsing and a `--stream` CLI flag
MLCommons — Llama 2 70B benchmark; Inference v5.0; GPT-OSS/DeepSeek-R1 update; v6.0 results file	27 Mar 2024; 2 Apr 2025; 24 Mar 2026; 1 Apr 2026	the three latency anchors and the 520-row results file
Leviathan, Kalman & Matias — speculative decoding	30 Nov 2022	arXiv 2211.17192v1
Chen et al. — speculative sampling	2 Feb 2023	arXiv 2302.01318
Liu, Yu, Park, Stoica & Cheung — *Speculative Decoding: Performance or Illusion?*	31 Dec 2025	arXiv 2601.11580; the independent measurement
Google — *DiffusionGemma: 4x faster text generation*; `google/diffusiongemma-26B-A4B-it` model card	10 Jun 2026; card read 6 Sep 2026	the quality concession and the 15-row table
Artificial Analysis — DiffusionGemma providers	read 6 Sep 2026	zero providers, output speed N/A