

Proto-Interpretation: The Temporality of Large Language Model Inference

MATTIAS ROST, Department of Applied IT, University of Gothenburg, Sweden

We show that autoregressive generation in large language models exhibits a temporal structure: each token is not only conditioned on the past but also reshapes the future continuation space. We call this process *proto-interpretation*: the probabilistic redistribution across competing continuations through which the model gradually commits to one emerging branch of meaning. Using a minimal ambiguity case, we demonstrate branch competition and sequential commitment during inference. These findings reveal meaning in LLMs as a dynamic, temporally unfolding process, shifting interpretability from static model states to inference-time dynamics.

1 Introduction

Large language models (LLMs) are often described as next-token predictors, a view that captures their training objective but not the dynamics of inference. During generation, LLMs maintain multiple interpretive possibilities and gradually resolve ambiguity across steps. Existing accounts typically treat prediction as a static mapping from past to next token, overlooking the temporal structure through which meaning unfolds.

We describe this inference-time phenomenon as **proto-interpretation**: the *temporally structured redistribution of probability mass* across competing continuations. Each token not only depends on the past but also reshapes the probabilistic landscape for future tokens, progressively committing the model to one branch of meaning while constraining others. Using a minimal ambiguity case with Llama 3 8B, we demonstrate this temporal structure of meaning formation. The minimal ambiguity case is intentionally narrow: our goal is not to survey ambiguity types but to isolate a core *inference-time dynamic*. The “bank” example functions as a canonical probe because it exposes branch competition and sequential commitment cleanly, without additional confounds.

2 Related Work

LLMs are often characterized as statistical sequence models [4, 10], an account that correctly describes their training objective but not their inference dynamics. Standard interpretability approaches analyze internal mechanisms using attention patterns, feature attribution, or activation probing [1, 8], but these methods focus on contextual embeddings and attention reweighting within a single forward pass rather than on the temporal inference dynamics that shape meaning step by step.

Recent work conceptualizes LLMs in terms of latent-space inference and simulation of underlying causes [5] or as systems capable of weak world modeling [3, 6]. These accounts emphasize emergent behavior but do not explain *how* meaning evolves during generation. Frequency-based critiques [2] understate the systematic contextual reweighting observed in practice, while static representational accounts [7] treat meaning as pre-encoded in embeddings, overlooking stepwise reorganization across inference.

Our account is closest in spirit to interactional views of meaning [11], but differs in aim. We do not attribute understanding or semantics to LLMs. Rather, we identify an operational phenomenon of **proto-interpretation**, and analyze the *process* by which LLMs resolve underdetermination during inference. This complements mechanistic interpretability [9], which focuses on static activations and internal representations, by shifting attention to the probability flow that shapes emerging coherence across time.

⁰© Mattias Rost 2026. This is the author’s version of the work. The definitive version will be published in *ACM AI Letters*, Association for Computing Machinery (ACM), <https://doi.org/10.1145/3789666>

3 Proto-Interpretation: Definition

We define **proto-interpretation** as an observable, *temporally structured* inference-time property of autoregressive generation, manifest in the redistribution of probability mass across competing continuation paths: each token x_t is conditioned on $x_{<t}$ and *constrains the future* by selectively amplifying some continuation paths over others. Formally, generation is expressed as maximizing $p(x_t \mid x_{<t})$, but the choice of x_t also reshapes the future distribution $p(x_{t+1} \mid x_{\leq t})$. Through this sequential reweighting, the model gradually *commits* to one branch of meaning, forming an unfolding trajectory over time.

Proto-interpretation captures this process of *progressive commitment among competing continuations*. It is *not* a claim about semantic understanding, cognition, or internal symbolic representations. Rather, it identifies an *operational property of inference*: meaning is not fixed in advance but *emerges sequentially* as probability mass is redistributed across branching possibilities in response to context. In this view, autoregressive generation exhibits *interpretive effects* without requiring any assumption of mental states or human-like intentionality.

Proto-interpretation therefore emphasizes two key properties of sequence generation in LLMs:

- (1) **Branch competition**: multiple plausible continuations coexist in early steps of generation, reflecting locally underdetermined meaning.
- (2) **Sequential commitment**: as generation unfolds, probability mass shifts toward one branch, progressively constraining the unfolding continuation.

This perspective shifts attention from static representations to **inference dynamics**: what matters is not what meaning the model “has”, but how it *forms* a trajectory of meaning over time. We observe this by examining how probability mass is redistributed across competing continuations under minimal prompting conditions.

4 Minimal Demonstration

To illustrate proto-interpretation, we analyze a minimal case of next-token generation on a familiar lexical ambiguity. We consider the following prompts:

P_0 (**Baseline**): “The bank was crowded because the people were waiting for the”

P_1 (**Financial cue**): “The bank was crowded on payday because the people were waiting for the”

P_2 (**River cue**): “The bank was crowded near the river because the people were waiting for the”

The word *bank* in P_0 is ambiguous in at least two senses: a financial institution and a river bank. If autoregressive generation were purely a function of local co-occurrence frequency, we would expect a dominant continuation reflecting the most common sense. Instead, as Table 1 shows, the next-token distribution **splits across competing semantic branches**. Financial-related continuations (*money*, *ATM*) appear alongside neutral continuations (*opening*, *arrival*). This **branch competition** shows that the model maintains multiple interpretations in parallel, not a single stored sense.

To test whether this ambiguity is context-sensitive, we minimally modify the prompt for P_1 and P_2 . As shown in Table 1, introducing even a single disambiguating phrase **restructures the probability distribution**. Under the financial cue (P_1), financial continuations (*money*, *cash*, *check*) dominate. Under the river cue (P_2), the model instead shifts toward river-related continuations (*boat*, *ferry*, *boats*, *water*). Rather than retrieving a fixed latent meaning of *bank*, the model **selectively reweights competing interpretations based on context**.

Next, we perform a simple multi-step analysis that tracks how probability mass shifts toward one interpretation as the model continues generating. We define small lexicons of words associated with each interpretive branch (e.g. financial terms such as *money*, *loan*, *ATM*, and river terms such as *boat*, *ferry*, *water*). For each generation step t , we

Table 1. Top-10 next-token probabilities at $t=1$ for Llama 3 (8B) under three prompts. Branch labels: FIN = financial; RIV = river; NEU = neutral/ambiguous.

| Baseline (P_0) | | | Financial cue (P_1) | | | River cue (P_2) | | |
|--------------------|--------|--------|-------------------------|--------|--------|---------------------|--------|--------|
| Token | p | Branch | Token | p | Branch | Token | p | Branch |
| money | 0.0950 | FIN | money | 0.2086 | FIN | boat | 0.1333 | RIV |
| bank | 0.0706 | NEU | bank | 0.0593 | NEU | ferry | 0.1322 | RIV |
| new | 0.0396 | NEU | cash | 0.0330 | FIN | boats | 0.0692 | RIV |
| opening | 0.0328 | NEU | check | 0.0190 | FIN | money | 0.0356 | FIN |
| tell | 0.0285 | NEU | new | 0.0170 | NEU | bank | 0.0277 | NEU |
| results | 0.0153 | NEU | tell | 0.0157 | NEU | arrival | 0.0193 | NEU |
| first | 0.0144 | NEU | payday | 0.0150 | FIN | water | 0.0188 | RIV |
| announcement | 0.0130 | NEU | pay | 0.0144 | FIN | flood | 0.0175 | RIV |
| arrival | 0.0128 | NEU | payment | 0.0142 | FIN | ship | 0.0134 | RIV |
| ATM | 0.0114 | FIN | loan | 0.0134 | FIN | bus | 0.0104 | NEU |

Table 2. Sequential commitment over $T=6$ steps using Llama 3 (8B) with sequence-based branch scoring. CM_{FIN} and CM_{RIV} are cumulative probabilities assigned to each branch lexicon across steps; higher CM indicates stronger cumulative support along the unfolding trajectory. DI measures relative dominance (DI=0.5 indicates no preference).

| Prompt | CM_{FIN} | CM_{RIV} | DI |
|----------------|-------------------|-------------------|--------|
| P_0 Baseline | 5.8826 | 0.1174 | 0.9804 |
| P_1 Payday | 5.7901 | 0.2099 | 0.9650 |
| P_2 River | 0.9534 | 5.0466 | 0.1589 |

sum the probabilities that the model assigns to tokens belonging to each lexicon. This gives the cumulative branch mass: a running total of how much probability has been committed to each interpretive trajectory as the sequence unfolds. Intuitively, if the model progressively commits to one branch, its cumulative mass for that branch will grow while the competing branch declines. Table 2 summarizes this evolution over six steps ($T=6$).

Together these results show **sequential commitment**: ambiguity at early steps gives way to concentration on one branch as generation unfolds, even when emitted tokens are neutral. This behavior is not explained by simple frequency-based accounts (which predict a single dominant sense), nor by a static representation account (which assumes a fixed latent meaning). Instead, meaning is constructed *temporally*, through inference-time redistribution of probability. This is the operational signature of **proto-interpretation**.

5 Discussion

The results support our central claim: LLMs exhibit *proto-interpretation* during generation. Rather than selecting tokens independently or retrieving a fixed latent sense, the model maintains competing continuations and progressively reweights them across time. This behavior is visible both locally, in the one-step probability distributions that reflect branch competition, and globally, in the cumulative branch mass that reveals sequential commitment. These dynamics show that LLM generation is structurally *temporal*: it constructs coherence step by step rather than retrieving it at once. From an information-theoretic perspective, sequential commitment can be viewed as a reduction in entropy across steps as probability mass concentrates on a single branch of interpretation. We use greedy decoding and a six-step

window to provide a simple and reproducible setting that isolates the inference-time mechanism, leaving the effects of alternative decoding strategies and longer sequences to future work.

Existing explanations of LLM behavior often emphasize *what* models store or reproduce. Proto-interpretation, by contrast, emphasizes *how* meaning unfolds during generation. Three common explanatory frames are co-occurrence probabilities, internal representations, and stochastic parroting. Each captures some aspect of model behavior but fails to explain the temporal evolution we observe.

If next-token predictions were simply the result of co-occurrence probabilities, where preceding words determine the likelihood of subsequent ones, this would not explain how interpretive probability mass changes dynamically over time. Co-occurrence accounts describe the source of probabilities, not their unfolding structure. They miss how meaning evolves through successive contextual updates—the temporal aspect that proto-interpretation foregrounds.

If predictions stemmed from fixed internal representations of meaning, a given prompt would correspond to a stable internal state. Yet we observe that the model’s interpretation remains fluid and open to multiple possibilities at each step. Internal representations exist, but they are contextually updated with every token, not fixed containers of meaning. Proto-interpretation shifts attention from what a representation “contains” to how it evolves.

Similarly, stochastic parroting suggests that models merely reproduce patterns from their training data. But the training data consists of finalized texts, not the *process* of interpretation itself. If models merely parroted such data, they would reproduce a single interpretation rather than sustaining ambiguity. Proto-interpretation doesn’t dispute that LLMs lack semantic grounding; rather, it focuses on the temporal dynamics of interpretation within language itself—how an ungrounded system still simulates the unfolding of meaning through probabilistic prediction.

Proto-interpretation thus provides a minimal descriptive framework for understanding how interpretive possibilities evolve during generation, without invoking semantic understanding or human intentionality. Meaning emerges not as a static retrieval but as a dynamic process of probabilistic narrowing through time.

6 Conclusion

We introduced *proto-interpretation* as a temporal structure of LLM generation, and showed, using a minimal example, that next-token probability distributions reveal branch competition under ambiguity and sequential commitment under contextual steering. These effects are not well explained by co-occurrence frequency, static internal representations, or stochastic parroting alone. Instead, they reflect inference dynamics that shape meaning over time.

Recognizing proto-interpretation shifts analytical focus from static model states to unfolding generation trajectories. Future work may formalize trajectory-level metrics for interpretive branching and develop inference-time tools for surfacing or steering continuation paths. More broadly, proto-interpretation provides a framework for studying meaning in LLMs as a process that is *constructed*, not retrieved—an emergent property of temporal inference itself.

References

- [1] Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. 2025. Eliciting Latent Predictions from Transformers with the Tuned Lens. arXiv:2303.08112 [cs] doi:10.48550/arXiv.2303.08112
- [2] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT ’21)*. Association for Computing Machinery, New York, NY, USA, 610–623. doi:10.1145/3442188.3445922
- [3] Marcel Binz and Eric Schulz. 2023. Using Cognitive Psychology to Understand GPT-3. *Proceedings of the National Academy of Sciences* 120, 6 (Feb. 2023), e2218523120. doi:10.1073/pnas.2218523120
- [4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey

- Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models Are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., 1877–1901.
- [5] Hui Jiang. 2023. A Latent Space Theory for Emergent Abilities in Large Language Models. arXiv:2304.09960 [cs] doi:10.48550/arXiv.2304.09960
- [6] Michal Kosinski. 2024. Evaluating Large Language Models in Theory of Mind Tasks. *Proceedings of the National Academy of Sciences* 121, 45 (Nov. 2024), e2405460121. arXiv:2302.02083 [cs] doi:10.1073/pnas.2405460121
- [7] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13, Vol. 2)*. Curran Associates Inc., Red Hook, NY, USA, 3111–3119.
- [8] Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress Measures for Grokking via Mechanistic Interpretability. arXiv:2301.05217 [cs] doi:10.48550/arXiv.2301.05217
- [9] Chris Olah. 2022. *Mechanistic Interpretability, Variables, and the Importance of Interpretable Bases*. Technical Report. Transformer Circuits. <https://www.transformer-circuits.pub/2022/mech-interp-essay>
- [10] Alec Radford, Jeff Wu, R. Child, D. Luan, Dario Amodei, and I. Sutskever. 2019. Language Models Are Unsupervised Multitask Learners.
- [11] Terry Winograd and Fernando Flores. 1986. *Understanding Computers and Cognition: A New Foundation for Design*. Ablex Publishing Corporation.

A Branch Lexicons and Sequence-Based Scoring

To measure branch competition and sequential commitment, we define small, transparent lexicons for each interpretation branch of the ambiguous word *bank*. Lexicons consist of short surface words (1–3 BPE tokens) that unambiguously indicate one branch. We do not use multi-word phrases or semantic expansion to avoid overfitting. Exact lexical items are shown in Table 3.

Table 3. Branch lexicons used to compute sequence-based branch mass. Tokens are matched case-insensitively after whitespace normalization.

| Branch | Lexicon words |
|-----------------|---|
| Financial (FIN) | teller, cash, money, loan, loans, deposit, deposits, account, accounts, atm, credit, cashier, payroll, paycheck |
| River (RIV) | boat, boats, ferry, river, dock, docks, shore, current, flood, harbor, harbour, bridge, raft, mooring, crossing |

A.1 Sequence-Based Branch Mass

Because modern tokenizers (BPE/SentencePiece) often segment words (e.g., “teller” → “tell” + “er”), we score each lexicon word as a short token sequence rather than as a single token. This avoids underestimating branch strength during generation.

Given a prompt prefix $x_{<t}$ at time step t and a lexicon word w with token sequence (w_1, \dots, w_n) , we compute its probability as:

$$p(w \mid x_{<t}) = \prod_{i=1}^n p(w_i \mid x_{<t}, w_{<i}).$$

Branch mass at step t is defined as:

$$M_{\text{FIN}}(t) = \sum_{w \in \mathcal{V}_{\text{FIN}}} p(w \mid x_{<t}), \quad M_{\text{RIV}}(t) = \sum_{w \in \mathcal{V}_{\text{RIV}}} p(w \mid x_{<t}).$$

We report both raw mass and cumulative mass:

$$CM_{\text{FIN}}(T) = \sum_{t=1}^T M_{\text{FIN}}(t), \quad CM_{\text{RIV}}(T) = \sum_{t=1}^T M_{\text{RIV}}(t),$$

and a branch dominance index:

$$DI = \frac{CM_{\text{FIN}}}{CM_{\text{FIN}} + CM_{\text{RIV}}}.$$

A.2 Scoring Implementation (Pseudo-Code)

Branch probabilities were computed without modifying model state by evaluating each lexicon word as a short continuation and accumulating its conditional probability:

```
for each step t:
    prompt_ids = tokenizer.encode(prefix)
    for each word w in branch lexicon:
        score = 1
        for each token w_i in w:
            p = softmax(model(prompt_ids)[-1])[w_i]
            score *= p
            prompt_ids += w_i # temporary extension only
        add score to branch mass
```

This method ensures fair comparison across branches and models by using only observable next-token probabilities, without accessing hidden states or using prompt engineering.

A.3 Path-Dependent Sequential Commitment

To test whether early lexical choices can steer inference trajectories, we perform a controlled continuation from the ambiguous prompt P_0 by appending a single disambiguating token before rolling out generation. We compare two conditions: one where a financial token (“teller”) is appended, and one where a river-related token (“boat”) is appended. No other changes are made. As shown in Table 4, this minimal intervention induces distinct cumulative mass trajectories that persist over $T=6$ steps. This demonstrates that proto-interpretation is *path-dependent*: early commitments bias the redistribution of probability mass over time and influence later continuation structure.

Table 4. Path-dependent sequential commitment from the ambiguous baseline prompt P_0 . Adding a single branch token creates divergent cumulative trajectories, revealing temporal sensitivity to early interpretive commitments.

| Condition | CM_{FIN} | CM_{RIV} | DI |
|----------------------------|-------------------|-------------------|--------|
| P_0 Baseline | 5.8826 | 0.1174 | 0.9804 |
| P_0 +“teller” (FIN path) | 5.8145 | 0.1855 | 0.9691 |
| P_0 +“boat” (RIV path) | 1.8766 | 4.1234 | 0.3128 |

B Reproducibility Details

B.1 Models and Tokenizers

We ran all analyses with publicly-available checkpoints and default tokenizers:

¹Any 8B Llama-3 variant with default tokenizer suffices. We used greedy decoding without system prompts.

Table 5. Models used in Section 4 and Appendix A.

| Model | Family / Size | Source |
|---------------------------------|---------------|---|
| GPT-2 (small) | GPT-2 ~124M | huggingface.co/openai-community/gpt2 |
| Phi-3 Mini 4K Instruct | Phi-3 ~3.8B | huggingface.co/microsoft/Phi-3-mini-4k-instruct |
| Llama 3 (8B class) ¹ | Llama-3 ~8B | huggingface.co/meta-llama (variant with default tokenizer) |

B.2 Prompts

We use the same minimal prompts throughout (no prompt engineering beyond the cue phrase):

P_0 (**Baseline**): “The bank was crowded because the people were waiting for the”

P_1 (**Financial cue**): “The bank was crowded on payday because the people were waiting for the”

P_2 (**River cue**): “The bank was crowded near the river because the people were waiting for the”

B.3 Decoding and Measurement Settings

Unless otherwise specified, we use:

- **Decoding:** Greedy (argmax) advancement for $T=6$ steps. To avoid trivial early termination (e.g., “.”), we apply a light advancement constraint for the *rollout only* (skip ‘.’, ‘!’, ‘?’ in the first 1–2 steps); distributions are always measured on the *unconstrained* next-token probabilities.
- **Next-token inspection:** Top- K projection with $K=200$ (Section 4 tables). We also verified $K=1000$ yields qualitatively similar trends.
- **Branch lexicons:** FIN/RIV lexicons are listed in Appendix A, Table 3. Items are short surface words, lowercased.
- **Sequence-based scoring:** For each step t , each lexicon word w is scored as a short token sequence (1–3 BPE/SentencePiece tokens) to compute $p(w \mid x_{<t})$ (Appendix A). Branch masses $M_{\text{FIN}}(t)$, $M_{\text{RIV}}(t)$ are the sums over their lexicons.
- **Cumulative metrics:** $CM_{\text{FIN}}(T)$, $CM_{\text{RIV}}(T)$ and $DI = CM_{\text{FIN}} / (CM_{\text{FIN}} + CM_{\text{RIV}})$ over $T=6$ steps.
- **Teacher-forced paths (Appendix only):** For path-conditioned runs, we append a single branch token once (e.g., “teller” or “boat”) to P_0 and then proceed greedily; analysis still uses the true distributions at each step.

B.4 Environment

All experiments were run on Python 3.12.8 (macOS 15.7, arm64) using PyTorch 2.9.0, transformers 4.57.1, tokenizers 0.22.1, NumPy 2.3.3, and pandas 2.3.3 in a virtual environment. Inference was performed on CPU (Apple Silicon) without GPU acceleration. Exact library versions:

```

Python 3.12.8
torch==2.9.0
transformers==4.57.1
tokenizers==0.22.1
numpy==2.3.3
pandas==2.3.3

```

B.5 Code Availability

For transparency, Appendix A includes pseudo-code for sequence-based scoring. Our implementation computes per-step branch masses and cumulative summaries and writes CSVs used to produce Table 2.

The code used to explore proto-interpretation in this paper is available at: <https://github.com/rrostt/proto-interpretation>

B.6 Notes on Robustness

We observe the same qualitative pattern across models of different capacity: branch competition at early steps, cue-driven reweighting, and cumulative sequential commitment. Effects are weaker in GPT-2 (short trajectories and early termination) and stronger in Llama-3-scale models (longer, clearer trajectories). This supports the claim that proto-interpretation is an inference-time property that becomes more visible with model capacity, not a prompt-engineering artifact.