Surface vs. Latent: Rethinking Language Representations

An Introduction to Dual-Stream Language Modeling
and the Case for Explicit Latent Structure

Bryan Leonard & Brandyn Leonard
Qira LLC, Maricopa, AZ

Abstract

Standard language models learn a single representation at each position that must simultaneously encode surface-level token statistics and deep structural patterns. We propose that language has distinct observable and hidden structures that benefit from explicit separation in the model architecture. The observable layer—surface—captures token-to-token patterns, local syntax, and factual associations. The hidden layer—latent—captures discourse mode, topic continuity, rhetorical structure, and long-range sequential dynamics. We describe how LOLM implements this separation through a Transformer surface decoder and a Mamba-style latent SSM, and present evidence that this explicit decomposition enables the model to learn representations that neither component could develop alone. Most strikingly, we show that removing the latent stream—which contributes only 30% of the gate weight at 1.57B—causes complete model collapse, while removing the surface stream produces degraded but functional language modeling.

1. The Single-Stream Assumption

Every major language model architecture—GPT, LLaMA, Mistral, Pythia, OPT—uses a single hidden state vector at each position. This vector is the model's entire representation of the input at that point. It must encode everything: the syntactic role of the current token, its semantic content, the broader topic of the passage, the rhetorical intent of the author, the discourse structure that determines what comes next, and any factual associations relevant to the context.

This is a remarkable compression. A single 2048-dimensional vector (at the 1.5B scale) must represent information that varies on timescales from single tokens ("which article precedes this noun?") to hundreds of tokens ("what genre is this document?"). The model's only mechanism for managing these different timescales is the depth of its layer stack and the breadth of its attention heads.

We propose that this compression is unnecessarily lossy, and that explicitly separating fast-changing surface representations from slow-changing latent representations yields better models.

2. What the Surface Stream Captures

The surface stream in LOLM is a standard pre-norm Transformer decoder with RoPE positional encodings. It does what Transformers do best: attend to specific prior tokens, compute syntactic and semantic relationships through multi-head attention, and build rich token-level representations through deep

feed-forward transformations.

Surface representations are fast-changing—they update significantly from token to token as the local context shifts. They excel at answering questions like: What is the grammatical structure of this sentence? What entity was mentioned three tokens ago? What word is most likely to continue this specific syntactic pattern? These are the bread and butter of language modeling, and the Transformer handles them with well-documented effectiveness.

3. What the Latent Stream Captures

The latent stream in LOLM is a 4-layer (at 304M) or 6-layer (at 1.57B) selective state-space model that takes the Transformer's hidden states as input and evolves a continuous latent representation z. Unlike the Transformer, the SSM maintains a recurrent state that evolves gradually—it naturally smooths over individual token variations and tracks slower-changing patterns.

The CPC training objective provides direct evidence of what the latent stream learns. By training the SSM to predict future decoder states from current latent states, we can measure whether the latent representation contains meaningful information about what the Transformer will encounter next. At 304M, the CPC loss drops from 5.55 (random chance over 256 positions) to 0.15—indicating that the latent state encodes highly precise information about the future trajectory of the surface decoder. The SSM is not simply smoothing the Transformer's output; it is extracting predictive structure that the Transformer does not explicitly represent.

4. The Seasoning Effect: Why 30% Matters More Than 70%

At convergence on the 1.57B model, the manifestation gate settles at approximately 0.71—meaning 71% of the fused representation comes from the surface decoder and 29% from the latent SSM. A naive interpretation would be that the surface decoder is doing the heavy lifting while the SSM provides a minor refinement.

The ablation data demolishes this interpretation. Removing the latent 29% (forcing gate to 1.0) causes perplexity to explode from 34.47 to 485 million—a 14 million-fold degradation. The surface decoder alone is not just worse; it has completely lost the ability to model language. Conversely, removing the surface 71% (forcing gate to 0.0) produces perplexity of 56,130—bad, but still functioning as a recognizable language model.

We call this the seasoning effect, by analogy with cooking: salt is a tiny fraction of a dish's ingredients by weight, but removing it destroys the result. The latent SSM provides a structural signal that the surface decoder has learned to depend on absolutely. Without it, the Transformer's representations become incoherent—not slightly worse, but catastrophically non-functional.

5. Why Explicit Separation Works

The key design principle is not simply "add an SSM alongside a Transformer." It is the explicit separation of training objectives and gradient paths. The surface decoder is trained by token prediction loss. The latent SSM is trained by CPC loss. The regime layer is trained by changepoint and diversity losses. The memory is trained through chunked read-write gradient bridging. Each component has its own learning signal, and

gradient isolation prevents the dominant token loss from overwhelming the auxiliary objectives.

This separation is what enables the components to develop complementary representations rather than collapsing to redundancy. The SSM is not trying to minimize token prediction loss directly—it is trying to predict future decoder states. The regime layer is not trying to minimize perplexity—it is trying to identify coherent segments. Each component solves its own problem, and the fusion layer learns to combine their outputs in a way that serves the overall objective.

6. Implications for Language Model Design

The surface-latent decomposition suggests a broader design principle: language models may benefit from explicitly representing distinct types of structure through dedicated subsystems, rather than forcing a single representation to encode everything. The current single-stream paradigm succeeds because Transformers are powerful enough to implicitly learn multiple timescales of representation within a single vector. But "powerful enough" is not the same as "optimal." The explicit separation in LOLM achieves 52% lower perplexity than Pythia-410M with 26% fewer parameters at 304M, and 25.9% lower perplexity than a matched baseline at 1.57B. These gains come precisely from giving each type of structure its own pathway and its own training signal.

The most compelling evidence that this matters is the dependency inversion. If the surface decoder could implicitly capture everything the latent SSM captures, removing the SSM at inference time would cause only moderate degradation. Instead, it causes complete collapse. The latent stream is carrying information that the surface stream cannot reconstruct from its own representations. Explicit separation didn't just improve efficiency—it enabled the model to learn representations that were previously inaccessible.

7. Conclusion

Language has structure at multiple timescales—from individual token statistics to document-level discourse patterns. Standard language models compress all of this into a single representation stream. LOLM demonstrates that explicitly separating surface and latent representations, training them with dedicated objectives, and combining them through a learned gate produces models that are more efficient, more capable, and internally organized in ways that single-stream models are not. The finding that the minority latent stream is more structurally essential than the majority surface stream challenges the assumption that language modeling is primarily a surface-level prediction task. It suggests that the hidden structure of language—the patterns that persist below the level of individual tokens—may be more fundamental to model quality than we previously understood.