Semantic Tensor Memory: Interpretable, Drift-Aware Memory for Tracking Meaning Across Time, Events, and Context

Joshua Farrow*[1]

[1]Independent Researcher / Applied AI Engineer, Richmond, VA, USA

December 2025

Abstract

We introduce *Semantic Tensor Memory* (STM), an interpretable external memory representation for longitudinal semantics. STM preserves event-level embeddings (e.g., token or atomic "semantic event" vectors) across ordered steps (conversation turns, clinical notes, dated documents), enabling measurement and explanation of semantic drift, inflection points, and concept evolution over time. Unlike flat vector stores that collapse text into chunk embeddings, STM makes *time* and *event granularity* first-class: each step stores a variable-length matrix of event embeddings, optionally paired with a sequence-level embedding for fast indexing and retrieval. We formalize STM as a *ragged tensor field* (a masked rank-3 substrate) with a principled extension to higher-order (N-D) memory via additional index modes (speaker, modality, source, model layer, resolution). We provide drift metrics spanning centroid trajectories, velocity/acceleration, and alignment-based event drift (assignment/transport), along with robust visualization methods (PCA/UMAP, heatmaps, trajectory graphs) designed for variable-length sequences. We discuss system architecture, storage/index design, governance considerations for clinical deployment, and applications in healthtech quality assurance, reflective agents, brand monitoring, and narrative-driven financial intelligence.

Keywords: external memory, semantic drift, concept evolution, interpretability, retrieval-augmented generation, longitudinal NLP, token embeddings, assignment alignment

1 Introduction

Large language models (LLMs) and modern decision-support systems increasingly require external memory: to maintain continuity across sessions, to ground responses in long-lived records, and to detect changes in user intent, clinical state, or market regime. Today, the dominant memory pattern is retrieval via vector databases: text is chunked, embedded, indexed, and retrieved by similarity. This is effective for static recall, but it compresses rich temporal structure into a bag of points and often obscures *how* meaning changes across time.

In high-stakes domains, *change* is the signal. Clinical notes drift as patient status evolves and teams shift their language. Organizations develop documentation and billing drift that predicts

*Correspondence: info@joshuafarrow.com

audit risk. Markets reprice narratives before fundamentals fully update. In these settings, we need memory that can answer not only: "What is relevant?" but "What has changed, when, and in what direction?"

We propose **Semantic Tensor Memory (STM)**, an interpretable memory substrate that preserves event-level embeddings over ordered steps. STM turns a session into a time-ordered collection of event embedding matrices (ragged by nature) plus optional sequence-level embeddings for indexing. STM supports:

- **Granular drift:** event alignment and drift heatmaps, not just chunk similarity.
- **Trajectory semantics:** centroid movement, velocity/acceleration, inflection-point cues.
- **Dual resolution:** store both event-level (fine) and step-level (coarse) representations to keep analysis explainable and retrieval fast.
- **N-D extensibility:** add modes (speaker, modality, source, model layer, temporal resolution) without losing a clean core.

2 Contributions

STM is a design pattern rather than a single learned model. This paper contributes:

1. **A formal memory object** for ordered text: STM as a ragged tensor field with explicit masking and optional batching.
2. **A practical architecture** for dual-resolution storage: event-level matrices + sequence-level embeddings + metadata.
3. **A family of drift metrics** spanning centroid drift, dynamics (velocity/acceleration), and alignment-based drift (assignment/transport).
4. **Robust visualization and diagnostics** for variable-length sequences that avoid silent truncation and report masking/validity.
5. **A principled answer to "3D vs N-D"**: STM is minimally rank-3 in its semantic substrate, but naturally extends to N-D via additional indexed modes.

3 Related Work

Vector stores and retrieval-augmented generation (RAG). RAG-style systems combine a parametric model with non-parametric memory by retrieving relevant documents at inference time [1]. This improves factuality and domain grounding, but typical implementations store only chunk-level vectors and treat memory as an unordered set.

Long-context and recurrence. Architectures such as Transformer-XL introduce recurrence to capture longer dependencies without unbounded context windows [2]. STM is complementary: it externalizes long-term structure into an inspectable memory that can be queried, summarized, and governed.

Memory-augmented neural networks. Differentiable external memory systems (e.g., DNC) demonstrate learnable read/write memory operations [3]. STM focuses on *user-facing interpretability* and system integration rather than end-to-end differentiable controllers.

Embeddings at multiple granularities. BERT provides contextual token embeddings [4]. Sentence-BERT provides efficient sentence/sequence embeddings suitable for similarity search [5]. STM leverages this split directly: fine-grained event embeddings for explanation and alignment; coarse embeddings for indexing and retrieval.

Dimensionality reduction and visualization. UMAP is widely used to visualize high-dimensional embeddings with good global structure preservation [6]. STM emphasizes *mask-aware* projections for ragged sequences and the reporting of invalid rows (NaN/Inf/padding).

Alignment-based comparisons. To compare two unordered sets of events (tokens), one can compute similarity matrices and solve assignment problems with the Hungarian method [7], or use optimal transport variants. STM uses alignment as an interpretable bridge: "what moved" becomes visible as matched event pairs.

4 Definitions and Formalism

4.1 Steps, events, and embeddings

Let a **step** be a time-ordered unit: a conversation turn, a dated note, a document window, or a fixed time slice. At step $t \in \{1, \dots, T\}$, let x_t be the raw text (or multimodal content). An **eventizer** maps x_t to a sequence or set of N_t events:

$$E_t = \{e_{t,1}, \dots, e_{t,N_t}\}.$$

In the simplest case, events are tokenizer subwords; in richer pipelines, events can be spans (entities, keyphrases), tool outputs, or structured fields.

An embedding model produces an event embedding in $\mathbb{R}^D$ for each event. We store the step as a matrix:

$$M_t \in \mathbb{R}^{N_t \times D}, \quad M_t[i, :] = \phi(e_{t,i}; \text{context}).$$

4.2 STM as a ragged tensor field (the "3D" substrate)

Conceptually, STM is a rank-3 object indexed by *time* t, *event* i, and *embedding channel* d. Because N_t varies, we represent STM as a *ragged tensor field*:

$$\text{STM}(x_{1:T}) \equiv \{M_t\}_{t=1}^{T}, \quad M_t \in \mathbb{R}^{N_t \times D}.$$

For batch computation, we materialize a padded tensor and mask:

$$\widetilde{M} \in \mathbb{R}^{T \times N_{\max} \times D}, \qquad \widetilde{W} \in \{0, 1\}^{T \times N_{\max}},$$

where $\widetilde{W}_{t,i} = 1$ iff event i is valid at step t. All analytics in this paper are defined to be *mask-respecting*.

4.3 Why it is "3D" and how it becomes N-D

The minimal semantic substrate needs (i) order, (ii) event granularity, and (iii) embedding channels. That yields the canonical rank-3 form. However, real systems require additional axes:

- speaker/role r (user/assistant/clinician)
- modality m (text/audio/vision/sensor)
- source/document stream s (news outlet, payer policy, client ID)
- embedding model/layer ℓ (multiple layers or multiple encoders)
- temporal resolution k (turn/day/week aggregation)

Rather than forcing a dense N-D tensor, STM treats these as *index modes* over a family of ragged slices:

$$M_t^{(m,r,s,\ell,k,\dots)} \in \mathbb{R}^{N_t \times D}.$$

Thus STM is "3D" in its core slice but naturally **N-D as a tensor-of-tensors** keyed by additional modes.

5 System Architecture

5.1 Dual-resolution memory

STM stores two coupled representations per step:

- **Event-level matrix** M_t (fine, explainable, alignment-ready).
- **Step-level vector** $z_t \in \mathbb{R}^{D_s}$ (coarse, indexable).

The step vector can be formed by pooling event embeddings (mask-aware mean) or by a separate sequence embedding model (e.g., Sentence-BERT) for fast approximate nearest-neighbor retrieval [5].

5.2 Context-conditioned embeddings

"Meaning" is contextual and temporal. STM supports lightweight contextualization:

$$\phi(e_{t,i};\, c_t) \quad \text{where} \quad c_t = g(\text{metadata}_t,\, x_{t-w:t-1}).$$

A simple and effective pattern is a **context prefix** p_t concatenated to text before embedding (for sequence embeddings), encoding domain hints, timestamps, or a rolling window summary. This can stabilize trajectories while preserving raw event embeddings for interpretability.

5.3 Storage schema (conceptual)

A practical STM store can be implemented atop files, a relational DB, or a vector DB with sidecar blobs. A minimal schema:

- `session(id, metadata_json)`
- `step(session_id, t, timestamp, raw_text, z_t, metadata_json)`
- `event(session_id, t, i, span_start, span_end, token, v_t,i, metadata_json)`

where $v_{t,i}$ stores rows of M_t. For scale, M_t can be stored as a compressed array (e.g., float16 + chunked compression) and loaded on demand.

6 Drift and Dynamics Metrics

We provide a toolkit; deployments select metrics based on cost, explainability, and sensitivity.

6.1 Step centroids and trajectory dynamics

Define the mask-aware centroid:

$$\bar{m}_t = \frac{1}{\sum_i \widetilde{W}_{t,i}} \sum_{i=1}^{N_{\max}} \widetilde{W}_{t,i}\, \widetilde{M}_{t,i,:}.$$

Define cosine drift between adjacent steps:

$$\Delta_t^{\text{cent}} = 1 - \cos(\bar{m}_{t-1}, \bar{m}_t).$$

With timestamps τ_t, define velocity and acceleration (discrete):

$$v_t = \frac{\bar{m}_t - \bar{m}_{t-1}}{\tau_t - \tau_{t-1}}, \qquad a_t = \frac{v_t - v_{t-1}}{\tau_t - \tau_{t-1}}.$$

These yield interpretable "semantic motion" and highlight inflection points (spikes in $\|v_t\|$ or $\|a_t\|$).

6.2 Alignment-based event drift (assignment)

Centroid drift can miss changes when overall meaning stays similar but constituent concepts rotate. For steps $t-1$ and t, build a similarity matrix:

$$S_{ij} = \cos\big(M_{t-1}[i,:],\, M_t[j,:]\big).$$

Convert to costs $C_{ij} = 1 - S_{ij}$. The **assignment drift** solves:

$$\pi^\star = \arg\min_{\pi \in \Pi} \sum_i C_{i,\pi(i)},$$

where Π is the set of one-to-one matchings between events (after optional truncation to a max length for cost control). The Hungarian method solves this in polynomial time [7]. Define:

$$\Delta_t^{\text{assign}} = \frac{1}{|\pi^\star|} \sum_i C_{i,\pi^\star(i)}.$$

This yields a **token/event drift heatmap**: matched pairs explain what persisted vs transformed.

6.3 Set-to-set drift variants

Depending on the deployment:

- **Mean-min drift:** for each event in t, find nearest neighbor in $t-1$.
- **Transport drift:** relax one-to-one matching (optimal transport) for repeated concepts.
- **Cluster transition drift:** cluster step-level vectors $\{z_t\}$ and examine regime transitions.

7 Visualization and Diagnostics

STM is only useful if it stays interpretable under real-world messiness.

7.1 Mask-aware projections

Given a set of event vectors (or centroids), compute PCA or UMAP on valid rows only and report:

$$\text{valid\_ratio} = \frac{\#\{\text{valid rows}\}}{\#\{\text{rows}\}}.$$

Project either:

- **All events:** show event clusters and how they shift per step.
- **Centroids:** show a trajectory through embedding space.

UMAP is recommended for non-linear structure; PCA provides axis interpretability [6].

7.2 Drift heatmaps and alignment maps

For T steps, compute a drift matrix $D \in \mathbb{R}^{T \times T}$ using centroid or assignment drift. Visualize:

- **Pairwise drift heatmap:** global view of regime shifts.
- **Adjacent-step alignment map:** token-to-token drift explanation.

7.3 Axis interpretation

To interpret PCA directions, select outlier events with extreme loadings and summarize them with lightweight heuristics or LLM-based labeling. The key constraint: axis labels must be grounded in *observable exemplars* (tokens/spans and example texts), not hallucinated narratives.

8 Retrieval and Use in LLM Agents

STM supports retrieval at multiple resolutions:

- **Step retrieval:** nearest neighbors in $\{z_t\}$ for fast RAG-style grounding [1].
- **Event retrieval:** find salient events matching a query embedding for explanation.
- **Trajectory retrieval:** retrieve segments with high acceleration (inflection windows).

A reflective agent can use STM in a loop:

1. Retrieve top-k relevant steps by z_t similarity.
2. Within each step, retrieve salient events (high similarity or high drift contribution).
3. Provide the LLM with *evidence* (verbatim text spans + event summaries) and *change signals* (drift metrics, inflection points).
4. Generate an answer plus a trace: "what changed" and "why this retrieval supports the claim."

9 Illustrative Example: ABA Session Notes

This section illustrates what STM makes visible. Consider a time series of ABA session notes for a young child. Each note is a step; events are tokens or extracted spans; embeddings produce M_t and z_t.

STM can reveal:

- **Trajectory:** a drift from "behavioral instability" clusters toward "skill acquisition" clusters.
- **Inflection points:** spikes in semantic velocity after intervention changes.
- **Alignment evidence:** disappearance of high-drift events (e.g., "elopement") and emergence of new stable concepts (e.g., "peer interaction").

Note. The above is an illustrative template; a full empirical section should report dataset details, preprocessing, model choices, and quantitative metrics (e.g., correlation with clinical outcomes, IOA changes, or audit findings).

10 Applications

10.1 Clinical QA and documentation governance

STM can detect documentation drift, measure consistency across clinicians/sites, and surface outlier wording that predicts compliance risk. Because STM exposes evidence at the event level, it supports auditability and clinician trust.

10.2 Financial market intelligence and narrative drift

STM can embed headlines, filings, and commentary as ordered steps, measuring theme rotation and regime changes. The key is not "sentiment" alone but *semantic transitions* (what the narrative becomes).

10.3 Brand monitoring and social listening

STM can track how customer language changes after releases or incidents, separating transient noise from sustained narrative shifts via drift dynamics.

10.4 Reflective agents and long-horizon autonomy

STM provides agents with an explicit memory of change, enabling "meta" reasoning: recognizing when user goals drift, when constraints change, and when old assumptions should be retired.

11 Governance, Privacy, and Safety

Token/event-level memory can expose sensitive information. High-stakes deployments require:

- **Minimization:** store only what is needed (consider span-level events rather than raw tokens).
- **Access control:** per-session and per-field policies; encrypt at rest.
- **Retention and redaction:** deletion workflows that remove both raw text and derived embeddings.

- **Auditability:** log retrieval queries, retrieved evidence, and downstream outputs.
- **Bias monitoring:** drift metrics can amplify spurious correlations; evaluate across cohorts/sites.

12 Limitations and Future Work

- **Embedding dependence:** STM inherits strengths/weaknesses of embedding models (domain shift, tokenization artifacts). BERT-style token embeddings are contextual but may not align with human semantic units [4].
- **Compute and storage:** event-level storage is heavier than chunk vectors. Compression, downsampling, and multi-resolution storage are important.
- **Alignment cost:** assignment/transport drift can be expensive; practical systems cap event counts or use approximate matching [7].
- **Evaluation:** the most important metrics are domain-grounded: predicting audits, outcomes, or user satisfaction—not just intrinsic clustering quality.
- **Multimodal STM:** extending events beyond text (audio/vision) is natural but requires careful eventization and alignment across modalities.

13 Conclusion

Semantic Tensor Memory reframes external memory from static recall to *interpretable semantic evolution*. By making time and event granularity first-class, STM supports drift-aware analysis, inflection detection, and explainable retrieval for real-world decision support and reflective agents. The core substrate is rank-3 (time $\times$ events $\times$ channels), but STM generalizes cleanly to N-D through indexed modes (speaker, modality, source, layer, resolution) without requiring brittle dense tensors.

Acknowledgments

Thanks to my wife, Elisa, and family, for their support and encouragement.

References

[1] P. Lewis et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *NeurIPS*, 2020. https://arxiv.org/abs/2005.11401

[2] Z. Dai et al. Transformer-XL: Attentive language models beyond a fixed-length context. In *ACL*, 2019. https://arxiv.org/abs/1901.02860

[3] A. Graves et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 2016. (Commonly referenced as the Differentiable Neural Computer.)

[4] J. Devlin et al. BERT: Pre-training of deep bidirectional transformers for language understanding. 2018. https://arxiv.org/abs/1810.04805

[5] N. Reimers and I. Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. 2019. https://arxiv.org/abs/1908.10084

[6] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. 2018. https://arxiv.org/abs/1802.03426

[7] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 1955.