Multi-Timescale Continuous Learning: Emergent Properties of Episodic Compression

Part 3 of Learning Through Narratives: Episodic Compression to Latent Variables for Sample-Efficient Intelligence

Terminology Note: As in Parts 1 and 2, we use terms like "learning," "understanding," and "reasoning" as behavioral shorthand without making claims about machine consciousness or internal mental states. When we say an AI system "learns" from narratives, we mean it produces behavioral responses consistent with narrative patterns. This usage aligns with our focus on behavioral competence rather than unverifiable internal states. Similarly, we use "latent variables" in the standard ML sense of learned hidden representations that capture underlying structure, distinct from LeCun's recent emphasis on continuous latent variables for handling aleatory uncertainty in predictions.

Abstract

Parts 1 and 2 established that episodic compression enables sample-efficient learning from narratives and proposed a two-layer architecture for scaling to real-world deployment. This discussion paper explores an emergent property of that architecture: it naturally operates across multiple timescales, from real-time pattern matching (seconds) through episodic formation (hours) to long-term world model accumulation (months-years). This temporal structure addresses a gap in current machine learning research, which focuses primarily on training efficiency rather than continuous learning during deployment. We examine implications for deployed AI systems, identify key distinctions from traditional continual learning (particularly regarding catastrophic forgetting), and outline open research questions. This work positions episodic compression not merely as an alternative learning mechanism, but as a practical architecture for systems that learn continuously over time while deployed.

1. Introduction

1.1 The Foundation

Parts 1 and 2 of this work established two key findings:

Part 1 demonstrated that episodic compression enables sample-efficient learning from narratives. A system with 9 generalizations and 4 episodes successfully predicted outcomes in novel scenarios within an artificial world (Shimmer Valleys), achieving appropriate confidence calibration and multi-step causal reasoning without requiring embodied interaction or millions of training examples.

Part 2 proposed a two-layer architecture for scaling episodic compression beyond toy demonstrations. Functional separation between Intuition (small model, large context, pattern-matching) and Executive (large model, standard context, reasoning and consolidation) addresses scaling challenges when accumulated knowledge exceeds single-context capacity.

These contributions challenged LeCun's assertion that embodiment is necessary for sample-efficient learning and provided a practical alternative implementable with current language models.

1.2 An Emergent Observation

While developing the two-layer architecture, we observed something unexpected: the system naturally operates across multiple timescales that mirror aspects of human learning:

- **Real-time** (seconds): Intuition pattern-matches current experiences against stored generalizations
- **Episodic formation** (minutes-hours): Salience detection identifies surprising events worthy of retention
- **Consolidation** (hours-days): Micro-sleep cycles extract patterns from accumulated episodes
- **Long-term accumulation** (weeks-months-years): Generalizations grow and refine during deployment

This temporal structure was not explicitly designed as a feature – it emerged from architectural decisions aimed at solving scaling challenges. Yet it addresses something current machine learning struggles with: continuous learning during deployment across multiple timescales.

1.3 The Deployment Learning Gap

Current machine learning operates in distinct phases:

Training phase (weeks-months): Process millions of examples, optimize weights, create frozen model

Deployment phase (indefinite): Model remains static, accumulates usage data

Retraining phase (occasional): Eventually retrain entire model on accumulated data

This paradigm has several limitations:

- Systems cannot adapt to deployment context without expensive retraining
- Each deployment remains generic rather than personalizing to specific use
- Knowledge accumulates as raw data rather than integrated learning
- Retraining risks catastrophic forgetting of previously learned capabilities
- Costs and complexity limit how often retraining occurs

Humans operate differently. We learn continuously throughout deployment (life), integrating new experiences with existing knowledge across multiple timescales without periodic complete retraining.

1.4 Scope and Claims

This paper explores implications of the episodic compression architecture for continuous learning during deployment. We make several claims:

1. The architecture naturally enables learning across multiple timescales
2. This addresses a gap in current ML regarding deployed learning
3. The learning is additive rather than destructive, fundamentally changing catastrophic forgetting
4. Several open research questions emerge from this temporal perspective
5. Practical benefits may accrue for deployed systems

We explicitly do not claim:

- That we have empirically validated long-term deployment (months-years of operation)
- That this is the only or best approach to continuous learning
- That all open questions have answers
- That this precisely mirrors human learning mechanisms
- That the architecture is ready for production deployment without further research

This is a discussion paper exploring implications, not a validation of a complete system.

1.5 Paper Organization

Section 2 analyzes how episodic compression operates across timescales. Section 3 examines the catastrophic forgetting problem and why this architecture addresses it differently. Section 4 discusses deployment learning and personalization. Section 5 identifies open research questions. Section 6 outlines future research directions. Section 7 concludes.

2. Multi-Timescale Operation

2.1 Human Learning Across Timescales

Before analyzing the architecture, consider how human learning operates temporally:

Immediate (seconds): Working memory holds current context, attention focuses on relevant information, pattern recognition identifies familiar structures.

Short-term (minutes-hours): Salient experiences enter conscious awareness, surprising events receive rehearsal, initial consolidation begins.

Medium-term (hours-days): Sleep consolidates memories, patterns extract from related experiences, connections form with existing knowledge.

Long-term (weeks-months-years): Integrated world models develop, hierarchical abstractions form, expertise builds through accumulated pattern recognition.

This multi-timescale structure appears fundamental to human learning efficiency. We don't process every experience equally – salience determines what receives consolidation effort. We don't consolidate constantly – periodic sleep cycles batch-process recent experiences. We don't store raw experiences indefinitely – compression into patterns enables efficient long-term storage.

2.2 Episodic Compression Timescales

The episodic compression architecture exhibits analogous temporal structure, though we make no claims about mechanistic similarity to biological learning:

Real-Time Operation (Seconds)

When new experiences arrive, both Intuition and Executive receive them simultaneously – mirroring how human cognition processes experiences through parallel pathways:

```
Experience: "Chrome flutter seed touches resonator"

Intuition (parallel, immediate):
- Scans all generalizations in large context window
- Scans stored episodes for direct matches
- Pattern matches in sub-second:
  * Generalizations: G1 (contact transfer), G5 (property transfer), G8
(resonator response)
  * Episodes: No direct episode match (too novel)
- Surfaces "gut feel": "This feels like contact-based property transfer"
- Duration: <1 second (small model, fast inference)

Executive (parallel, deliberate):
- Receives raw experience details
- Receives Intuition's pattern matches (gut feel)
- Reasons: "Given gut feel about contact transfer, chrome's reflective
property might transfer to resonator..."
- Generates prediction informed by Intuition's matches
- Duration: Several seconds (complex reasoning)
```

This parallel processing mirrors human cognition: you feel a "gut reaction" (Intuition's pattern-match) before consciously understanding why (Executive's deliberate reasoning). The gut feel arrives first and guides where conscious attention focuses.

Dual memory matching: Intuition matches against both generalizations (abstract patterns) and episodes (specific memorable cases). Episode matches often produce stronger, faster reactions because they are more concrete. This enables the system to handle "X generally but not in this specific case" scenarios.

Example – The Bob Problem:

- Generalization: "Firemen are helpful and trustworthy"
- Episode (high salience): "Bob the fireman was rude at that party"
- New situation: Bob appears at event
- Episode match fires: Strong immediate "avoid Bob" reaction
- But generalization still applies: Welcome other firemen

Without stored episodes, the system could only reason at the generalization level, losing the specific concrete cases that drive strong gut reactions. This mirrors biological episodic memory (specific autobiographical events) versus semantic memory (general knowledge), both informing human cognition (Tulving, 1972).

Customer support example: Query arrives about login issues. Intuition immediately:

- Generalization match: "This fits authentication failure patterns"
- Episode match: "Reminds me specifically of User X's session timeout issue"
- Executive receives both matches and reasons: "Given this is similar to User X's case and matches auth patterns, likely session timeout. Solution: clear cache and re-login."

The small model size (1-3B parameters) enables Intuition's fast inference. Large context window (100K-200K tokens) enables comprehensive matching across entire knowledge base (both generalizations and exceptional episodes). Both happen continuously during operation.

Episodic Formation (Minutes-Hours)

As experiences unfold, salience detection identifies which warrant retention:

Type 1 salience (prediction mismatch): Executive makes prediction based on generalizations, outcome violates prediction, experience flagged as high-salience episode.

Type 2 salience (novelty): Intuition pattern-matches and finds no adequate matches, signals novelty to Executive, experience flagged for retention.

Episodes accumulate in a buffer awaiting consolidation. Most experiences do not become episodes – only those marked by surprise or novelty. This selectivity provides the first level of compression.

Consolidation Cycles (Hours-Days)

When episode accumulation reaches threshold (suggested 5-10 episodes) or maximum time elapses (suggested 2-6 hours), consolidation triggers:

Micro-sleep duration: 10-60 seconds typically **Frequency during active learning**: 3-8 times per day **Frequency during mature operation**: Less frequent as fewer surprises occur

During consolidation:

Executive phase (pattern extraction, generalization, and curation):

1. Creates episodes from flagged high-salience experiences
2. Queries Intuition: "Which generalizations match these episodes?"
3. Identifies patterns across episodes (new + historically relevant)
4. Forms new generalizations or updates existing ones
5. Performs consistency checking
6. **Prunes newly created episodes aggressively** - keeps only highly exceptional episodes (~10% of created episodes) that have teaching value as specific memorable cases alongside abstract generalizations
7. Writes filtered episodes (only exceptional cases) plus new/updated generalizations to Intuition

Intuition phase (autonomous memory management): 8. Receives pre-filtered content from Executive (generalizations and exceptional episodes, already curated for novelty/value) 9. Manages its store autonomously over time based on:

- Usage patterns (what's being accessed?)
- Age (how old is this content?)
- Context pressure (approaching window limits?)
- Contribution to active patterns

10. Can archive or forget content based on these factors
11. Applies simple policies without expensive reasoning

This creates two-stage filtering: Executive determines what's worth keeping at all (novelty/teaching value), while Intuition manages what stays active versus archived (resource management). The dual storage of generalizations and exceptional episodes mirrors biological distinctions between semantic memory (general knowledge) and episodic memory (specific autobiographical events), both informing cognition. Generalizations remain most stable – Intuition would archive hundreds of episodes before considering archiving a single generalization.

Long-Term Accumulation (Weeks-Months-Years)

Over extended deployment, generalizations accumulate and mature:

Week 1: Initial generalizations form from first surprising experiences, tentative confidence levels, rapid learning as everything is novel.

Month 1: 50-200 generalizations accumulated, confidence strengthens for reliable patterns, fewer surprises as common patterns are covered.

Month 3-6: 200-500 generalizations, hierarchical structure may emerge, meta-patterns about pattern types, increasingly sophisticated prediction.

Year 1+: 500-1000+ generalizations, deep domain expertise for deployment context, personalized world model specific to this use case, mature confidence calibration.

This long-term trajectory represents continuous learning during deployment – the system becomes increasingly adapted to its specific operational context without requiring parameter retraining.

2.3 Timestamping as the Minimal Temporal Mechanism

The system's multi-timescale behavior – seconds for pattern-matching, hours for consolidation, months for cumulative expertise – emerges naturally from explicit timestamping rather than from any specialized temporal module. Each episode and generalization carries a single timestamp. For episodes, this marks when the system experienced the event. For generalizations, this marks when the pattern was formed or last updated. These timestamps enable the architecture to measure recency, detect drift, and evaluate temporal spacing between experiences. No separate scheduler or temporal reasoning engine is required – the timestamps are simply part of each stored artifact.

Over continuous operation, this simple mechanism yields distinct temporal strata:

Immediate time: milliseconds-to-seconds sequencing during ongoing interaction

Episodic time: minutes-to-hours intervals defining when new experiences are consolidated

Developmental time: weeks-to-months accumulation of generalizations and confidence changes

Because timestamps require no external scheduler or graph structure, they provide the minimal substrate for emergent temporal organization – allowing time to be learned through experience rather than imposed through design.

2.4 Why These Timescales Matter

The temporal structure provides several benefits:

Computational efficiency: Real-time pattern-matching uses small models (fast, cheap). Expensive consolidation happens periodically in batches (amortized cost). Long-term accumulation has no ongoing training costs (no GPUs, no gradient descent).

Salience filtering: Not every experience requires consolidation. Surprise-driven selection focuses learning effort where it matters. Reduces noise from routine, expected experiences.

Consolidation quality: Batch processing enables cross-episode pattern detection. Single episodes might show spurious patterns. Multiple related episodes enable robust generalization.

Progressive learning: System adapts gradually over deployment timeline. Each consolidation refines the world model incrementally. Avoids disruption of wholesale retraining.

Context adaptation: Different deployments encounter different experience distributions. Learning during deployment enables personalization to specific context. Generic pre-training + specific accumulated knowledge.

2.4 Comparison with Current ML Temporal Structure

Traditional machine learning has different temporal characteristics:

Training:

- Duration: Days to weeks of continuous processing
- Scale: Millions to billions of examples
- Mechanism: Gradient descent optimizing loss function
- Output: Frozen model weights

Deployment:

- Duration: Indefinite
- Learning: None (weights frozen)
- Adaptation: None without retraining
- Data accumulation: Raw logs, not integrated knowledge

Retraining:

- Frequency: Occasional (weeks-months between cycles)
- Risk: Catastrophic forgetting of previous capabilities
- Cost: Full training expense repeated
- Disruption: New model may break existing use cases

Episodic compression operates differently:

- Consolidation: Minutes to hours per cycle (not days/weeks)
- Frequency: Multiple times per day during operation
- Mechanism: Pattern extraction through reasoning (not gradient descent)
- Learning: Continuous during deployment
- Adaptation: Immediate integration of new patterns
- Risk: Additive learning reduces catastrophic forgetting (more in Section 3)

The temporal profile better matches operational needs: frequent small updates rather than infrequent massive retraining.

2.5 Implications for Deployed Systems

This temporal structure suggests several practical implications:

Deployment trajectory: Systems improve over time in deployment. Week 1 performance < Month 1 < Month 6. Value increases with operational duration. Creates switching costs (accumulated knowledge specific to context).

Personalization without effort: Each deployment learns its specific context automatically. No manual customization or fine-tuning required. Generic model + specific experiences = personalized system.

Reduced training costs: No periodic retraining on GPUs. Consolidation uses inference (cheaper than training). Long-term cost structure more favorable.

Faster adaptation: New situations integrate within hours-days via consolidation. No waiting for next training cycle. Responsive to changing conditions.

Observable learning: Generalizations accumulate visibly over time. Can track system's learning progress. Interpretable knowledge growth vs. opaque weight changes.

These implications remain somewhat speculative – comprehensive long-term deployment studies are needed to validate these benefits empirically.

2.6 Neurological Parallels and Architectural Validation

The functional separation between Intuition and Executive, and their parallel processing of experiences, finds interesting parallels in neurological case studies:

Damage to older brain structures (basal ganglia, cerebellum): Patients lose automatic pattern recognition and procedural memory. They struggle to recognize familiar patterns – no "gut feel" guides their responses. They can still reason deliberatively about situations, but everything requires conscious effort. They "live in the moment" without accumulated pattern recognition informing behavior.

Architectural analog: If Intuition were damaged or lost, the system would have Executive reasoning capability but no gut feel pattern recognition. Every situation would require deliberate reasoning from first principles. No immediate sense of "this feels familiar" to guide analysis. The system could handle novel situations but would be inefficient – like a human having to consciously think through every familiar task.

Damage to prefrontal cortex and hippocampus: Patients retain old routines and procedural responses. They have gut feel about familiar situations and can respond appropriately to established patterns. However, they struggle with novel situations requiring flexible reasoning or adaptation when situations deviate from established patterns.

Architectural analog: If Executive were damaged or lost, the system would have Intuition's pattern-matching capability and gut feel but no ability to reason beyond established patterns. It could immediately recognize "this matches pattern G7" but couldn't adapt when the situation requires combining patterns in novel ways or reasoning about unprecedented combinations.

Parallel processing validation: Both systems receive experiences simultaneously. Old brain structures provide rapid pattern recognition (gut feel) while prefrontal cortex performs slower deliberate reasoning. The gut feel arrives first – often within milliseconds – before conscious analysis completes. This guides where conscious attention focuses.

The architecture mirrors this: both Intuition and Executive receive raw experiences directly. Intuition's sub-second pattern-matching provides gut feel. Executive's deliberate reasoning (taking seconds) uses this gut feel to guide analysis. The Executive doesn't wait for complete reasoning before receiving Intuition's input – it gets the gut feel immediately and reasons informed by it.

These neurological cases validate several architectural decisions: pattern recognition and deliberate reasoning are genuinely separable functions that can be damaged independently, both process experiences in parallel, gut feel arrives before deliberate reasoning completes, and both components are necessary for full capability.

3. Catastrophic Forgetting Reconsidered

3.1 The Traditional Catastrophic Forgetting Problem

Continual learning research focuses extensively on catastrophic forgetting: when neural networks learn new tasks, fine-tuning overwrites weights optimized for previous tasks, destroying previously learned capabilities (McCloskey & Cohen, 1989; French, 1999).

Example: Model trained on medical diagnosis, then fine-tuned on legal analysis, loses medical diagnostic capabilities.

Why it occurs: Neural network weights encode all knowledge. Training adjusts weights to minimize loss on current data. No explicit mechanism prevents overwriting previous knowledge. Gradient descent optimizes current task, ignoring previous tasks.

Current solutions: Elastic Weight Consolidation (protect important weights), Progressive Neural Networks (add new capacity), rehearsal methods (mix old data with new), memory replay (store examples from previous tasks).

These approaches mitigate catastrophic forgetting but don't eliminate it. The fundamental issue remains: knowledge encoded in weights creates interference between tasks.

3.2 Episodic Compression Learning Mechanism

Episodic compression operates through a fundamentally different mechanism:

Knowledge representation: Generalizations stored as structured text in Intuition's context, not as neural network weights. Each generalization is a discrete symbolic entity.

Learning operation: Consolidation adds new generalizations to the store or updates existing generalizations' confidence/scope. Does not overwrite neural network parameters. Pre-trained models remain unchanged.

Update mechanism:

```
New experience → Episode formation → Consolidation extracts pattern →
  IF similar generalization exists:
    Update confidence (strengthen or weaken)
    Refine scope (add conditions/exceptions)
    Split into variants if contradictory evidence
  ELSE:
    Add new generalization to store
```

This is **additive learning**: new knowledge adds to or refines existing knowledge rather than overwriting it.

3.3 Why Catastrophic Forgetting Doesn't Apply

Traditional catastrophic forgetting assumes:

1. Knowledge encoded in neural network weights
2. Learning modifies those weights
3. Finite capacity creates interference

Episodic compression counters these assumptions:

Assumption 1 counter: Knowledge not in weights. Generalizations stored symbolically in context. Pre-trained model weights never change during deployment learning.

Assumption 2 counter: Learning adds generalizations, doesn't modify weights. Consolidation is reasoning over episodes, not gradient descent. New patterns integrate without weight updates.

Assumption 3 counter: Storage capacity is context window, not parameter count. Context windows accommodate thousands of generalizations. Expansion possible by using larger context models.

Result: The catastrophic forgetting problem as traditionally formulated doesn't apply to this architecture.

3.4 Different Forgetting Challenges

While catastrophic forgetting in the traditional sense doesn't occur, episodic compression faces different forgetting challenges. Importantly, forgetting happens at two stages with different criteria:

Stage 1 - Executive pruning (during consolidation): Executive aggressively prunes newly created episodes, keeping only those with exceptional novelty or teaching value (~10% typically). Most episodes are discarded immediately after their patterns extract into generalizations. Only exceptional episodes survive to be written into Intuition's store as specific memorable cases.

Filtering criterion: "Is this worth keeping as a specific memorable case, or have I captured its essence in generalizations?" Based on novelty, uniqueness, teaching value as concrete example.

Stage 2 - Intuition management (autonomous, ongoing): Intuition manages its store over time based on usage patterns, age, context pressure, and contribution to active generalizations. Can archive or forget content that Executive previously deemed valuable but which has become less relevant.

Filtering criterion: "Should this stay active or be archived?" Based on resource management, access patterns, operational relevance.

Forgetting hierarchy:

1. **Most episodes** (discarded by Executive immediately): After pattern extraction, most episodes never reach Intuition
2. **Surviving episodes** (managed by Intuition): Even valuable episodes may be archived by Intuition over time based on usage
3. **Generalizations** (most stable): Last to be archived, as they represent refined distilled knowledge

This two-stage filtering reflects different purposes: Executive curates for value, Intuition manages for resources.

Challenge 1: Executive pruning criteria

Intuition receives pre-filtered content (Executive already pruned aggressively). Over time, Intuition must manage this store based on simple policies. What factors should govern archiving decisions?

Potential factors:

- Age-based: Archive content older than threshold
- Usage-based: Archive content not accessed recently
- Contribution-based: Episodes supporting active generalizations stay active
- Context pressure: When approaching limits, archive lowest-priority content first

Since Intuition uses a small model, policies must be simple and not require expensive reasoning. But what's the right balance between different factors? How should policies adapt as the system matures?

Challenge 3: Context window limits

Even with aggressive Executive pruning and Intuition management, accumulated content may eventually exceed large context windows. Intuition must decide what stays immediately accessible versus what gets archived.

Potential solutions:

- Hierarchical organization (high-level patterns + drill-down detail)
- Importance-based retention (frequently-used patterns in context, rare ones archived)
- Domain clustering (activate relevant subsets based on current context)

This is a **resource management problem**, not catastrophic forgetting – old knowledge isn't destroyed, just moved to longer-term storage.

Challenge 4: Contradictory evidence

New experiences may contradict established generalizations. Executive must decide during consolidation: update the generalization, split into conditional variants, or reject new evidence as outlier.

This is a **belief revision problem**, not catastrophic forgetting. Old knowledge isn't destroyed – it's refined or qualified based on new evidence.

Example resolution:

```
Original: "Contact with globs transfers color" (HIGH confidence)
New evidence: Contact with phase glob doesn't transfer color
Resolution: Split into variants:
  - "Contact with normal globs transfers color" (HIGH confidence)
  - "Contact with phase globs does not transfer color" (MEDIUM confidence)
```

Challenge 3: Outdated knowledge

Deployment contexts change. Patterns that were true become false. Old generalizations may mislead if world rules shift.

This is a **temporal validity problem**, not catastrophic forgetting. Knowledge was correct, context changed. Solutions involve:

- Tracking generalization age and context
- Detecting performance degradation over time
- Pruning or archiving outdated patterns

3.5 Advantages of Additive Learning

The additive learning mechanism provides several advantages:

Transparent knowledge: Generalizations are human-readable. Can inspect what system has learned. Can manually remove or modify problematic patterns. Debugging and correction more tractable.

Controlled forgetting: Explicit decisions about what to retain or prune. Not side effect of training, but deliberate policy. Can implement domain-specific retention strategies.

Graceful degradation: If context limits reached, can archive less-critical generalizations. System still operates with reduced capacity. Doesn't catastrophically forget everything.

Belief revision: New evidence updates beliefs explicitly. Can track confidence changes over time. Conditional reasoning handles contradictions naturally.

Auditable evolution: Can trace how generalizations formed and changed. Complete history of learning available. Enables accountability for system behavior.

3.6 Open Questions

Several questions remain about forgetting in episodic compression systems:

Forgetting policy: What principles should govern what to forget? Age-based? Usage frequency? Confidence level? Domain relevance?

Forgetting timing: When should pruning occur? During consolidation? Separate maintenance cycles? Triggered by performance metrics?

Forgetting scope: Individual episodes? Entire generalizations? Both? How to decide?

Recovery from errors: If system forms incorrect generalization, how to correct? Manual intervention? Automated detection? What safeguards needed?

These questions require empirical investigation with deployed systems operating over extended periods.

4. Deployment Learning and Personalization

4.1 The Generic-to-Specific Trajectory

Episodic compression systems follow a trajectory from generic pre-trained capabilities to context-specific expertise:

Day 1: Deployment initialization

- System: Pre-trained language model + initial world model
- Initial world model: Either empty or seeded with curated domain knowledge
- Capabilities: Generic reasoning, no context-specific knowledge
- Prediction quality: Relies entirely on pre-training

Week 1: Initial adaptation

- Experiences: Everything is novel, high salience
- Episodes: 20-50 formed from surprising events
- Consolidations: Frequent (daily or more)
- Generalizations: 10-30 formed about deployment context
- Learning rate: Rapid (many surprises)

Month 1: Context understanding

- Experiences: Common patterns recognized, fewer surprises
- Episodes: 50-150 accumulated
- Generalizations: 50-200 covering major patterns
- Prediction quality: Substantially improved for common scenarios
- Learning rate: Moderate (some novelty remains)

Month 3-6: Mature operation

- Experiences: Most situations covered by existing patterns
- Episodes: Slow accumulation (mainly edge cases)
- Generalizations: 200-500+ comprehensive coverage
- Prediction quality: High for deployment context
- Learning rate: Slow (only novel situations trigger learning)

Year 1+: Deep expertise

- Generalizations: 500-1000+ fine-grained patterns
- Capabilities: Deep context-specific expertise
- Personalization: Highly adapted to specific deployment
- Value: Substantial switching costs (accumulated knowledge)

This trajectory is qualitative projection – actual timelines depend on experience diversity and domain complexity.

4.2 Personalization Without Configuration

Traditional systems require explicit personalization:

- Manual configuration of parameters
- Custom training on user-specific data
- Integration with user's existing systems
- Ongoing maintenance as needs change

Episodic compression enables implicit personalization:

The system learns deployment context automatically through operation. No explicit personalization effort required. Each deployment accumulates its own unique generalization set. Same base system + different experiences = different specialized systems.

Example: Customer support deployment

Company A (B2B SaaS, technical users):

- Episodes: Complex technical issues, API integration problems
- Generalizations: "When error X occurs with integration Y, solution Z works"
- Accumulated knowledge: Deep technical troubleshooting patterns
- Personality: Highly technical, detailed solutions

Company B (Consumer product, non-technical users):

- Episodes: UI confusion, billing questions, common misunderstandings
- Generalizations: "When user reports X, usually they meant Y"
- Accumulated knowledge: Translation from user language to actual issues
- Personality: Simplified explanations, patient clarification

Same initial system, different accumulated knowledge through deployment learning.

4.3 Continuous Adaptation

Deployment contexts change over time:

- Products evolve with new features
- User populations shift
- Competitive landscape changes
- Regulations update
- Seasonal patterns emerge

Static systems: Require periodic retraining to stay current. Gap between changes and model updates. Risk breaking existing capabilities with retraining.

Episodic compression: Adapts continuously as context changes. New patterns integrate within hours-days of emergence. Old patterns update or qualify as contradictions emerge. No disruption from wholesale retraining.

Example adaptation scenario:

Month 1: E-commerce system learns shipping delay patterns

- Generalization: "Orders to region X typically arrive in 3-5 days"

Month 3: New warehouse opens, changes delivery patterns

- New evidence: Orders now arriving in 1-2 days
- Consolidation: Updates generalization confidence, splits by time period
- Result: "Before Date D: 3-5 days. After Date D: 1-2 days"

Month 6: Seasonal surge causes delays

- New evidence: December orders taking 5-7 days
- Consolidation: Adds seasonal qualifier
- Result: "Normal: 1-2 days. December: 5-7 days"

System adapted three times without manual intervention or retraining. Knowledge refined incrementally as context evolved.

4.4 Network Effects and Lock-In

Episodic compression's continuous learning creates interesting business dynamics:

Value increases over time: Month 1 system < Month 6 system < Year 1 system. Same deployment becomes more valuable with duration. Customer gets better system without additional payment.

Switching costs accumulate: Accumulated generalizations are deployment-specific. Switching to competitor means starting from zero. Lost expertise creates friction for churn.

Data network effects: More usage → more episodes → better generalizations → better performance → more usage. Positive feedback loop creates defensibility.

Personalization moat: Each deployment is unique to its context. Competitors cannot replicate accumulated knowledge. Generic competitors less valuable than adapted system.

These dynamics favor the provider who can deploy continuously learning systems, though we acknowledge this analysis remains speculative without real-world validation.

4.5 Implications for System Design

Deployment learning suggests several design principles:

Design for learning: Optimize system for accumulating knowledge during operation, not just performing tasks. Surface learning progress to users. Make generalization formation visible and controllable.

Encourage exploration: Users benefit from exposing system to diverse scenarios early. Early novelty → more episodes → faster generalization formation. Create incentives for users to "teach" the system.

Manage expectations: Set realistic expectations for Day 1 vs. Month 6 performance. Communicate learning trajectory explicitly. Help users understand system is improving.

Provide transparency: Show users what system has learned. Display generalizations, confidence levels, supporting episodes. Enable manual correction of incorrect patterns.

Enable knowledge management: Let users curate learned knowledge. Archive outdated generalizations. Import/export knowledge bases. Share learnings between related deployments.

5. Open Research Questions

The episodic compression architecture and its temporal properties raise numerous questions requiring investigation:

5.1 Consolidation Dynamics

Optimal scheduling: What triggers should activate consolidation? Pure time-based? Pure salience-based? Hybrid? Does optimal schedule vary by domain or change as system matures?

Consolidation depth: Should all consolidations be equal, or should some be deeper/longer? Might fast consolidations handle immediate patterns while periodic deep consolidations restructure world model?

Computational budget: How much processing should consolidation consume? Trade-off between consolidation quality and operational cost? Adaptive budgeting based on episode importance?

Success metrics: How to evaluate consolidation quality? Prediction accuracy improvement? Generalization count stability? Episode compression ratio?

5.2 Generalization Formation

Threshold sensitivity: Proposed 3-5 episode threshold for generalization formation – is this optimal? Does it vary by pattern complexity or confidence requirements? Can systems learn their own thresholds?

Pattern detection algorithms: Currently relies on Executive's reasoning capabilities – are there more systematic approaches? Graph-based pattern matching? Formal causal inference? Statistical significance testing?

Confidence calibration: How should confidence evolve with evidence? Bayesian updating? Frequency-based? Multi-factor scoring? How to avoid over-confidence from limited data?

Generalization scope: When to form broad vs. narrow generalizations? How to identify appropriate abstraction level? When to split overly broad patterns into conditional variants?

5.3 Memory Management and Forgetting

The dual memory structure (generalizations and exceptional episodes) complicates forgetting strategies beyond simple hierarchies.

Traditional view assumes: Episodes are raw material, generalizations are refined knowledge, therefore forget episodes before generalizations.

Dual memory reveals: Some episodes are more valuable than their generalizations. The "Bob the fireman" episode produces stronger gut reactions and handles exception cases that generalizations cannot capture. Exceptional episodes that contradict or override generalizations may be MORE valuable than many routine generalizations.

Usage-based forgetting: Rather than category-based hierarchy (episodes < generalizations), forgetting should consider:

- **Episode usage:** Frequently-matched episodes prove their value through access patterns. Exceptional episodes (Bob cases) that override generalizations merit retention even if rarely accessed. Episodes fully captured by generalizations and never accessed become candidates for archiving.
- **Generalization usage:** Frequently-applied patterns with successful predictions remain valuable. Never-applied patterns or those consistently failing predictions become archival candidates.
- **Relationship dynamics:** Episodes supporting frequently-used generalizations (concrete grounding) warrant retention of both. Episodes contradicting generalizations (exception handling) are especially valuable. Orphaned content without usage or connections suggests lower utility.

Human parallel: People remember vivid exceptional episodes (traumatic events, violations of expectations, emotionally salient moments) more than routine experiences, even when general patterns are well-established. Memory is not organized by simple hierarchies but by salience, utility, and emotional weight.

Open questions requiring research:

- What metrics best predict episode utility? (access frequency, salience score, exception value?)
- How should Intuition's autonomous policies balance different factors? (usage, age, contribution, salience)
- When do exception episodes (Bob cases) warrant indefinite retention versus eventual archiving?
- How do forgetting policies interact with learning - does aggressive episode pruning impair future generalization refinement?
- Can systems develop adaptive forgetting strategies based on their operational patterns?

5.4 Race Conditions and Override Mechanisms

The parallel processing architecture creates timing asymmetries between Intuition and Executive that mirror human cognitive dynamics.

The race condition: When experiences arrive, Intuition's pattern-matching (<1 second, small model) completes before Executive's deliberate reasoning (2-5 seconds, complex analysis). This creates situations where gut feel impulses emerge before reasoned judgments.

Example - Friend with snake: Friend approaches holding pet snake. Intuition immediately matches snake pattern → FEAR/DANGER response → impulse to avoid. Executive receives both raw experience and Intuition's fear signal, then reasons about context (trusted friend, controlled setting, social cues of safety) → decision to override fear impulse. Result: visceral fear feeling that conscious reasoning suppresses.

Biological parallel: LeDoux (1996) identified dual pathways for emotional processing - a fast "low road" (thalamus → amygdala, ~100ms) producing immediate emotional responses, and a slower "high road" (thalamus → cortex → amygdala, ~200-500ms) enabling cortical regulation. The emotional response occurs before conscious appraisal completes. The architectural race condition mirrors this established neural timing.

Design implications requiring investigation:

- **Override mechanisms:** How should Executive override or modulate Intuition's impulses? What information flow enables effective suppression without losing the signal value of gut feelings?
- **Confidence reporting:** Intuition might signal high confidence in pattern match (strong fear pattern, 90% match) while Executive's reasoning reduces confidence in danger (given context, 20% actual threat). Final confidence must reflect reasoned judgment, not just pattern match strength. How to communicate this progression transparently?

- **Streaming responses:** If systems generate responses incrementally, might output Intuition's initial assessment before Executive override completes. Could manifest as "This appears dangerous... but given the context, it's actually safe." Very human-like but requires careful design. How to handle gracefully?
- **Observable behavior:** Users might see systems "change their mind" in real-time as Executive overrides initial gut reactions. This mirrors human cognition ("At first I was scared, but then I realized...") but requires validation that users find this natural rather than concerning.
- **When does Intuition win?** Reflex-speed responses (catching falling objects, startle reactions) occur before Executive completes processing. Deliberate responses allow Executive override. What determines the threshold? How to design for appropriate timing?

This race condition appears to be a feature rather than bug: Fast pattern recognition enables rapid responses when speed matters. Deliberate reasoning enables override when context demands. The tension between them produces adaptive behavior matching human cognitive dynamics. However, systematic research is needed to validate these design intuitions and develop principled approaches to managing the race condition in deployed systems.

5.4 Hierarchical Organization

Meta-patterns: Can systems form generalizations about generalizations? "When facing causal reasoning problems, combining these 3 pattern types usually works"?

Abstraction levels: How to organize knowledge hierarchically? Domain → subdomain → specific pattern? Abstract → concrete? Temporal scales?

Dynamic reorganization: As knowledge accumulates, when to restructure organization? Triggered by size thresholds? Performance degradation? Proactively?

Cross-cutting concerns: Some patterns apply across domains – how to represent these efficiently without duplication?

5.5 Transfer and Multi-Agent Learning

Domain transfer: How do patterns learned in one domain transfer to related domains? Which generalizations are domain-specific vs. domain-general?

Knowledge sharing: Multiple deployments in related contexts – can they share learned patterns? What's appropriate granularity? How to handle conflicting evidence between agents?

Collective learning: Could episodic compression enable new forms of distributed learning? Agents learn locally, periodically synchronize generalizations? Federated episodic learning?

Teaching and explanation: Can systems explain their generalizations to humans or other systems? Transfer knowledge through narrative rather than just sharing generalization objects?

5.6 Safety and Robustness

Adversarial manipulation: Can attackers form incorrect generalizations through strategic episode injection? What defenses exist? Anomaly detection? Consensus mechanisms?

Bias accumulation: If deployment context has biases, system learns those biases. How to detect? Mitigate? Distinguish harmful bias from legitimate context-specific patterns?

Correction mechanisms: When system learns wrong pattern, how to correct efficiently? Manual intervention? Automated detection? What granularity of correction needed?

Verification: How to verify learned knowledge is correct? Testing regimes? Formal methods? Human review? Combination?

5.7 Empirical Validation

Long-term studies: Deploy systems for months-years, measure actual learning trajectories. Do predictions about continuous improvement hold? What unexpected behaviors emerge?

Domain diversity: Test across diverse domains to understand generality. Which findings are universal? Which are domain-specific? Where does approach work best?

Comparative studies: Compare against baselines (static models, fine-tuning, other continual learning approaches). What are actual advantages and disadvantages? Cost-benefit analysis?

Failure mode analysis: What failure modes emerge at scale? How do systems degrade? What causes learning to plateau or break?

6. Future Research Directions

Based on open questions, we identify several research directions:

6.1 Empirical Deployment Studies

Priority: High (validates fundamental claims)

Approach: Deploy episodic compression systems in real-world contexts for extended periods. Measure learning curves, generalization accumulation, prediction accuracy evolution, user satisfaction, operational costs. Compare against static baselines and periodic retraining approaches.

Suggested contexts:

- Customer support (multiple companies, track adaptation)
- Technical troubleshooting (measure expertise development)

- Domain consulting (compare to human learning curves)
- Content moderation (adaptation to evolving guidelines)

Key questions: Does continuous learning deliver sustained value? What are actual cost curves? How do failure modes manifest? What user behaviors emerge?

6.2 Consolidation Optimization

Priority: Medium-high (affects efficiency and quality)

Approach: Systematically vary consolidation parameters (frequency, depth, triggers) and measure effects on learning quality, operational costs, convergence speed. Develop principled approaches to scheduling.

Investigations:

- Time-based vs. salience-based vs. hybrid triggers
- Adaptive scheduling based on learning phase
- Deep vs. shallow consolidation cycles
- Computational budget allocation strategies

Key questions: What scheduling strategies optimize quality/cost trade-offs? Do optimal parameters vary by domain? Can systems learn their own schedules?

6.3 Hierarchical Knowledge Organization

Priority: Medium (important for scaling)

Approach: Develop mechanisms for hierarchical generalization organization. Investigate meta-pattern formation, abstraction levels, cross-cutting concerns. Measure effects on retrieval efficiency and reasoning quality.

Investigations:

- Automatic hierarchy formation algorithms
- Meta-generalization detection and formation
- Cross-domain pattern identification
- Dynamic reorganization triggers and mechanisms

Key questions: How to organize knowledge as it scales to thousands of generalizations? When do meta-patterns emerge? What are efficient representations?

6.4 Memory Management Research

Priority: Medium (becomes critical at scale)

Approach: Investigate the two-stage filtering process: Executive pruning during consolidation and Intuition's autonomous management over time. Develop principled approaches for both stages. Measure effects on learning quality, prediction accuracy, and resource efficiency.

Investigations:

- Executive pruning criteria: What makes an episode valuable enough to keep?
- Intuition policy development: Age-based, usage-based, contribution-based, or hybrid?
- Two-stage interaction: How should Executive and Intuition coordinate?
- Generalization archiving: Under what conditions should even generalizations move to cold storage?
- Archive/restore mechanisms: Efficient storage and retrieval strategies
- Resource monitoring: How to detect when policies need adjustment?

Key questions: What principles should govern Executive's aggressive pruning? What simple policies enable Intuition to manage effectively without expensive reasoning? How do the two stages interact optimally? Can systems develop self-tuning policies adapting to their operational patterns?

6.5 Multi-Agent and Transfer Learning

Priority: Medium-low (extends basic capabilities)

Approach: Investigate knowledge sharing between agents and transfer between domains. Develop protocols for collective learning and safe knowledge exchange.

Investigations:

- Peer-to-peer generalization sharing protocols
- Domain transfer mechanisms and limitations
- Collective intelligence through distributed learning
- Conflict resolution between agent knowledge bases

Key questions: Can episodic compression enable new forms of collective learning? What are boundaries of knowledge transfer? How to ensure safety?

6.6 Safety and Verification

Priority: High (required for deployment)

Approach: Develop adversarial testing regimes. Create verification methods for learned knowledge. Design correction mechanisms for incorrect generalizations.

Investigations:

- Adversarial episode injection attacks and defenses

- Bias detection and mitigation strategies
- Automated verification approaches
- Human-in-the-loop correction workflows

Key questions: How robust is episodic compression to manipulation? What verification approaches scale? How to enable safe correction?

6.7 Theoretical Foundations

Priority: Medium (supports other work)

Approach: Develop formal frameworks for episodic compression. Investigate theoretical properties of additive learning, convergence guarantees, sample complexity bounds.

Investigations:

- Formal models of episodic compression dynamics
- Convergence analysis for generalization formation
- Sample complexity bounds for different domains
- Relationship to existing learning theory

Key questions: What are theoretical properties of this learning mechanism? Under what conditions does it converge? What are efficiency guarantees?

7. Conclusion

Parts 1 and 2 established that episodic compression enables sample-efficient learning from narratives and proposed a practical architecture for scaling to real-world deployment. This discussion paper explored an emergent property of that architecture: it naturally operates across multiple timescales, enabling continuous learning during deployment.

The temporal structure mirrors aspects of human learning: real-time pattern matching (seconds), episodic formation from salient events (minutes-hours), periodic consolidation extracting patterns (hours-days), and long-term world model accumulation (weeks-months-years). This addresses a gap in current machine learning, which focuses primarily on training efficiency rather than deployed learning dynamics.

Several key insights emerge from this temporal perspective:

Catastrophic forgetting reconsidered: Traditional catastrophic forgetting assumes knowledge encoded in neural network weights that get overwritten during learning. Episodic compression uses additive learning – generalizations stored symbolically, new knowledge adds rather than overwrites. While different forgetting challenges emerge (context limits, outdated knowledge, memory management), the traditional problem doesn't apply.

Deployment learning trajectory: Systems follow predictable trajectories from generic capabilities to context-specific expertise. Week 1 rapid learning as everything is novel. Month 1 understanding common patterns. Month 3-6 mature operation with comprehensive coverage. Year 1+ deep expertise personalized to deployment context. This creates value that increases over time and switching costs that accumulate with duration.

Personalization without configuration: Each deployment automatically learns its specific context through operation. Same base system + different experiences = different specialized systems. No manual personalization effort required. Continuous adaptation as contexts evolve.

Research opportunities: The architecture opens numerous research questions about consolidation dynamics, generalization formation, forgetting mechanisms, hierarchical organization, transfer learning, safety, and long-term empirical validation. These questions represent productive directions for future work.

Practical implications: If empirical validation confirms theoretical benefits, episodic compression may enable a new class of deployed AI systems that improve continuously during operation, adapt automatically to changing contexts, and accumulate personalized expertise without expensive retraining cycles.

However, we emphasize important limitations. This remains largely theoretical – comprehensive empirical validation at scale awaits future research. The proposed mechanisms require testing across diverse domains and extended operational periods. Optimal parameters need empirical tuning. Safety considerations require careful investigation before production deployment.

The relationship to LeCun's JEPA framework remains complementary. JEPA emphasizes learned representations through embodied interaction and gradient descent optimization. Episodic compression emphasizes symbolic knowledge accumulation through narrative learning and consolidation reasoning. Both address sample efficiency through different mechanisms. JEPA may prove superior for sensorimotor tasks requiring embodied grounding. Episodic compression may prove superior for abstract reasoning tasks where knowledge exists in narrative form.

The deeper contribution of this work may be recognizing that learning happens across timescales and that current ML architectures address this poorly. Whether episodic compression specifically proves to be the best solution, the temporal perspective on learning deserves more attention in machine learning research. Systems that can learn continuously during deployment, across multiple timescales, adapting to specific contexts while maintaining coherent world models – these capabilities appear increasingly important as AI systems move from research laboratories into sustained real-world operation.

Future work must validate these ideas empirically, refine the mechanisms through practical deployment, address open questions systematically, and explore alternative approaches to multi-timescale learning. This discussion paper aimed to articulate the temporal dimension of episodic compression and invite research community engagement with these questions.

References

Core Framework

1. LeCun, Y. (2022). A path towards autonomous machine intelligence. Open Review, 62.
2. Mossrake Group, LLC. (2025a). Learning Through Narratives: Episodic Compression to Latent Variables for Sample-Efficient Intelligence. Part 1 of a series.
3. Mossrake Group, LLC. (2025b). Scaling episodic compression: A two-layer architecture for continuous learning. Part 2 of a series.

Episodic Memory and Learning

4. Tulving, E. (1972). Episodic and semantic memory. In E. Tulving & W. Donaldson (Eds.), Organization of memory (pp. 381-403). Academic Press.
5. Schacter, D. L., & Addis, D. R. (2007). The cognitive neuroscience of constructive memory: Remembering the past and imagining the future. Philosophical Transactions of the Royal Society B, 362(1481), 773-786.
6. Conway, M. A. (2009). Episodic memories. Neuropsychologia, 47(11), 2305-2313.
7. Ranganath, C., & Hsieh, L.-T. (2016). The hippocampus in time and space: Functional properties of episodic memory. Neuron, 90(2), 328–339.

Narrative Cognition

8. Bruner, J. (1986). Actual minds, possible worlds. Harvard University Press.
9. Bruner, J. (1990). Acts of meaning. Harvard University Press.
10. McAdams, D. P. (1993). The stories we live by: Personal myths and the making of the self. William Morrow.
11. Zacks, J. M., Speer, N. K., Swallow, K. M., Braver, T. S., & Reynolds, J. R. (2007). Event perception: A mind–brain perspective. Psychological Bulletin, 133(2), 273–293.
12. Kurby, C. A., & Zacks, J. M. (2008). Segmentation in narrative comprehension. Cognitive Psychology, 57(1), 25–61.

Curiosity and Intrinsic Motivation

13. Oudeyer, P. Y., & Kaplan, F. (2007). What is intrinsic motivation? A typology of computational approaches. Frontiers in Neurorobotics, 1, 6.
14. Schmidhuber, J. (2010). Formal theory of creativity, fun, and intrinsic motivation (1990-2010). IEEE Transactions on Autonomous Mental Development, 2(3), 230-247.

Sample Efficiency and Few-Shot Learning

15. Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. Science, 350(6266), 1332-1338.

16. Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. Behavioral and Brain Sciences, 40, e253.

Causal Learning

17. Pearl, J. (2009). Causality: Models, reasoning, and inference (2nd ed.). Cambridge University Press.
18. Gopnik, A., & Schulz, L. (2007). Causal learning: Psychology, philosophy, and computation. Oxford University Press.
19. Tenenbaum, J. B., Lake, B. M., Kemp, C., & Gershman, S. J. (2023). Compositional causal learning and time. Cognitive Science, 47(2), e13209.

Value Learning

20. Russell, S. (2019). Human compatible: Artificial intelligence and the problem of control. Viking.
21. Christiano, P., Leike, J., Brown, T. B., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. Advances in Neural Information Processing Systems, 30.

World Models

22. Ha, D., & Schmidhuber, J. (2018). World models. arXiv preprint arXiv:1803.10122.
23. Hafner, D., Lillicrap, T., Ba, J., & Norouzi, M. (2019). Dream to control: Learning behaviors by latent imagination. arXiv preprint arXiv:1912.01603.

Memory-Augmented Networks

24. Graves, A., Wayne, G., & Danihelka, I. (2014). Neural Turing machines. arXiv preprint arXiv:1410.5401.
25. Graves, A., Wayne, G., Reynolds, M., et al. (2016). Hybrid computing using a neural network with dynamic external memory. Nature, 538(7626), 471-476.

Meta-Learning

26. Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. International Conference on Machine Learning (ICML), 1126-1135.

Retrieval-Augmented Generation

27. Lewis, P., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 33, 9459-9474.

Cognitive Systems

28. Kahneman, D. (2011). Thinking, fast and slow. Farrar, Straus and Giroux.

29. Schneider, W., & Shiffrin, R. M. (1977). Controlled and automatic human information processing: I. Detection, search, and attention. Psychological Review, 84(1), 1-66.
30. Shiffrin, R. M., & Schneider, W. (1977). Controlled and automatic human information processing: II. Perceptual learning, automatic attending and a general theory. Psychological Review, 84(2), 127-190.

Catastrophic Forgetting

31. McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. Psychology of Learning and Motivation, 24, 109-165.
32. French, R. M. (1999). Catastrophic forgetting in connectionist networks. Trends in Cognitive Sciences, 3(4), 128-135.

Continual Learning

33. Kirkpatrick, J., et al. (2017). Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences, 114(13), 3521-3526.
34. Rusu, A. A., et al. (2016). Progressive neural networks. arXiv preprint arXiv:1606.04671.

Cognitive Architecture

35. Schmidhuber, J. (2010). Formal theory of creativity, fun, and intrinsic motivation (1990–2010). IEEE Transactions on Autonomous Mental Development, 2(3), 230–247.
36. Ha, D., & Schmidhuber, J. (2018). World Models. arXiv:1803.10122.
37. Hafner, D., Lillicrap, T., Ba, J., Fischer, I., & van den Oord, A. (2019). Learning latent dynamics for planning from pixels. Proceedings of ICLR.
38. Graves, A., Wayne, G., & Danihelka, I. (2014). Neural Turing Machines. arXiv:1410.5401.
39. Graves, A., Wayne, G., Reynolds, M., Harley, T., et al. (2016). Hybrid computing using a neural network with dynamic external memory. Nature, 538, 471–476.

Memory and Learning

40. Tulving, E. (1972). Episodic and semantic memory. In E. Tulving & W. Donaldson (Eds.), Organization of Memory (pp. 381–403). Academic Press.
41. Conway, M. A. (2009). Episodic memories. Neuropsychologia, 47(11), 2305–2313.
42. Schacter, D. L., & Addis, D. R. (2007). Constructive memory: The ghosts of past and future. Nature, 445(7123), 27–30.
43. Ranganath, C., & Hsieh, L.-T. (2016). The hippocampus in time and space: Functional properties of episodic memory. Neuron, 90(2), 328–339.

Temporal Modeling and Multi-Timescale Learning

44. Dai, Z., Wang, Y., & Dong, L. (2024). Transformer architectures for event time modeling. Proceedings of ICLR 2024.

45. Ahuja, A., & Singh, A. (2023). Memory-augmented transformers for continual learning. Proceedings of ICML 2023.
46. Kazemi, S. M., Goel, R., Kipf, T., Kazemi, A., Brubaker, M., & Hamilton, W. L. (2023). Temporal knowledge graphs: A survey. Journal of Artificial Intelligence Research, 77, 1335–1385.
47. Botvinick, M., Wang, J. X., Dabney, W., Miller, K. J., & Kurth-Nelson, Z. (2019). Reinforcement learning, fast and slow. Trends in Cognitive Sciences, 23(5), 408–422.
48. Kiebel, S. J., Daunizeau, J., & Friston, K. J. (2008). A hierarchy of time-scales in the brain. PLoS Computational Biology, 4(11), e1000209.
49. Tallec, C., & Ollivier, Y. (2018). Can recurrent neural networks warp time? Proceedings of ICLR 2018.
