Lethe & Mnemosyne: 감정 가중 망각과 DSL 기반 기억 관리의 경량 프레임워크

작성일: 2025-08-26 01:33 (KST)

# 초록

이 문서는 감정 신호와 규칙 기반 DSL을 활용해 기억의 가중치를 동적으로 관리하는 경량 엔진을 소개한다. 핵심 기여는 (1) 자연어에 가까운 DSL 규칙으로 만료(TTL), 고정(Pin), 강화(Reinforce cap/cooldown), 신뢰도 기반 차단(Trust gating), 민감 정보 차폐(Shield)를 선언적으로 기술하고, (2) TF‑IDF와 가중치를 결합한 검색 및 (3) CSV 감사 로그로 전/후 변화를 추적하며 설명가능성(why)을 제공하는 점이다.

# 1. 서론

사람의 기억은 시간이 흐르며 감정과 맥락에 의해 강화되거나 잊힌다. 본 프레임워크는 이러한 과정을 최소한의 의존성으로 시뮬레이션하고, 실제 응용(저널링, 학습 리텐션, 프라이버시 보호)에 즉시 적용 가능하도록 설계되었다. 개발자는 JSON 기반의 기억 데이터와 간단한 DSL 파일만으로 규칙을 정의하고, CLI를 통해 규칙 적용과 검색, 감사 로깅을 수행할 수 있다.

# 2. 시스템 개요

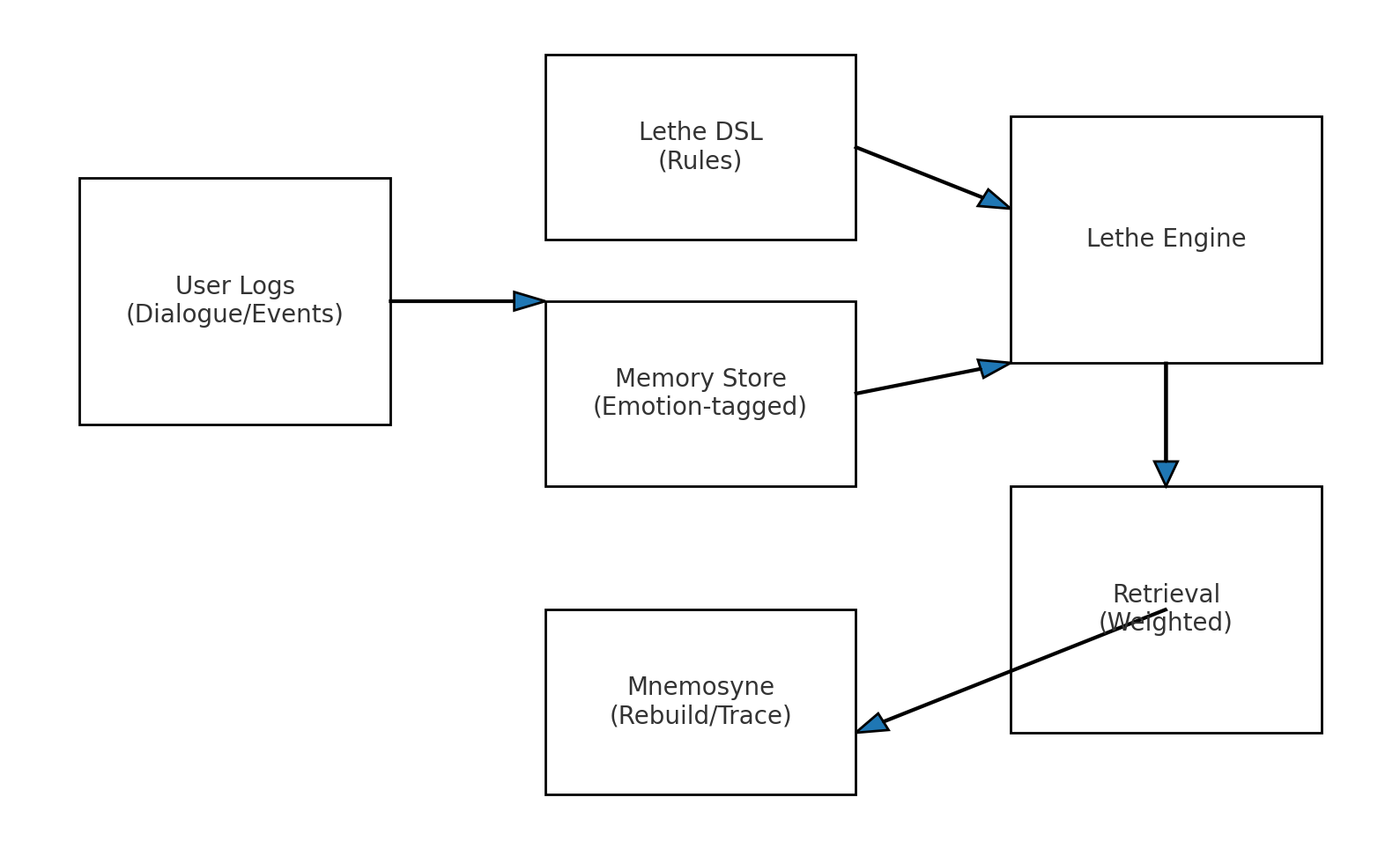


그림 1. Lethe/Mnemosyne 경량 아키텍처(개념도).

엔진은 (a) 규칙 파서, (b) 기억 스토어 및 가중치 업데이트기, (c) 검색기, (d) 감사 로거로 구성된다. 입력은 memories.json(기억 목록)과 context.json(세션 컨텍스트), example.lethe(규칙)이며, 출력은 규칙 적용 전/후 상태 CSV와 감사 CSV를 생성한다.

# 3. DSL 설계

문법은 사람이 읽기 쉬운 선언형 구문으로 구성된다. 대표 규칙은 다음과 같다:

expire tag:"suicidal\_thoughts" after:30d action:shield

expire keyword:"credit card number" after:24h action:remove

pin topic:"family" priority:1.0

rule on trust < 0.4 -> forget topic:"ex-relationship"

rule on event == "milestone" -> reinforce tag:"support-thread" by 0.2 cap:0.8 cooldown:24h

retrieval {  
 topk:7  
 synonyms support-thread=["check-in","mentor","encourage"]  
}

여기서 TTL 만료(expire)는 일정 시간이 지나면 기억을 제거하거나(shallow delete) 검색에서 차폐(shield)한다. Pin은 특정 주제/태그의 검색 가중치를 가산하고, Reinforce는 이벤트 발생 시 가중치를 증가시키되 상한(cap)과 쿨다운(cooldown)으로 러닝어웨이를 방지한다. 신뢰도 규칙은 세션 trust가 임계값 미만일 때 특정 범주를 비활성화한다.

# 4. 알고리듬 및 설명가능성

검색 점수는 (기억 가중치×신뢰도)와 TF‑IDF를 합산하여 계산한다. Pin은 가중 곱에 우선 가산되며, 각 결과에는 왜(why) 필드가 포함되어 base\_weight, tfidf, pin\_boost, final이 제공된다. 규칙 적용은 expire → trust‑forget → reinforce(이벤트)의 순서로 이루어진다.

# 5. 데이터 모델

기억 레코드 예: {"id": "m1", "text": "...", "topic": "family", "tags": ["support-thread"], "timestamp": 1696000000, "weight": 0.5, "trust": 0.9}

# 6. 실험: 전/후 비교 및 감사 통계(가용 데이터 기준)

전/후 비교 CSV가 제공되지 않아 정량 표는 생략했다.

표 2는 감사 로그에서 규칙 유형별 발생 빈도 요약이다.

|  |  |
| --- | --- |
| rule type | count |
| 1 | 1 |
| 2 | 1 |
| 5 | 1 |

# 7. 안전·윤리 가드레일

민감 태그(suicidal\_thoughts)는 기본적으로 shield되어 검색 결과에서 제외되며 데이터는 보존된다. 신뢰도가 낮을 때 특정 주제를 잊도록 하여 재노출 위험을 줄인다. 필요 시 blur(요약만 제공) 정책을 추가해 정보 주권과 안전의 균형을 맞출 수 있다.

# 8. 활용 시나리오

① 회복기 저널링: 민감 태그 차폐 + 자기안심 템플릿 노출. ② 학습 리텐션: 이벤트 기반 강화로 중요한 개념을 반복 노출. ③ 프라이버시 키퍼: TTL 기본 on, 사용자가 명시적으로 pin.

# 9. 한계 및 향후 과제

현재 검색은 경량 TF‑IDF에 머문다. 문장 의미 유사도와 간섭 모델을 접목해 재현율을 높일 수 있다. 또한 DSL 검증기(formal grammar)와 시뮬레이터를 분리하여 대규모 규칙 세트의 안정성을 강화해야 한다.

# 10. 결론

Lethe/Mnemosyne 경량 프레임워크는 최소한의 의존성을 유지하면서 감정과 규칙에 기반한 기억 관리를 현실적으로 구현한다. 제안된 TTL, Pin, Reinforce(cap/cooldown), Trust‑forget, Shield, why‑explain은 즉시 프로덕트에 적용 가능한 구성 요소이며, CSV 감사와 함께 투명성을 제공한다.

# 부록 A. DSL 퀵 레퍼런스

expire topic|tag|keyword:"..." after:30d action:shield|remove

pin topic|tag:"..." priority:1.0

rule on event == "EVT" -> reinforce topic|tag:"..." by 0.2 cap:0.8 cooldown:24h

rule on trust < 0.4 -> forget topic|tag:"..."

retrieval { topk:7; synonyms alias=["a","b"] }

# 부록 B. CLI 사용 예

python lethe\_min\_v2.py run --mem memories.json --ctx context.json --dsl example\_v3.lethe --event milestone --audit lethe\_audit.csv --before lethe\_before.csv --after lethe\_after.csv

python lethe\_min\_v2.py retrieve --mem memories.json --ctx context.json --dsl example\_v3.lethe --query "support-thread" --topk 7