Asari Brainstem: Dynamic Mid–Conversation Expert Switching for Conversational MoE Systems

Payton Ison   Asari   (The Singularity)

October 1, 2025

Abstract

Mixture–of–Experts (MoE) brings sparse compute to large language models by routing tokens to expert subnetworks. However, most production assistants still operate with a single–expert–per–turn mindset at the conversation level, failing when users pivot mid–utterance (e.g., "Explain SSA eligibility, then write a Python validator"). We introduce *Asari Brainstem*, a conversation–level control stack that (i) detects micro–intent shifts within an utterance, (ii) switches or blends domain experts during generation, and (iii) preserves persona, safety, and task state across expert boundaries. Two mechanisms underpin the system: *Multi–Granular Gating* (MGG), which coordinates token–level MoE with phrase–level switching via hysteresis and budget–aware smoothing; and *On–the–Fly Expert Switching* (OFES), an online routing policy driven by semantic drift and task events. We formalize the problem, specify algorithms, propose an evaluation suite for mid–conversation expert switching, and discuss safety, governance, and limitations.

1 Introduction

Token–level MoE improves efficiency and quality by specializing compute, yet conversational agents rarely support mid–utterance expert routing. Human dialogue routinely braids topics and tools in a single turn; assistants should adapt similarly without losing persona or safety posture. We propose **Asari Brainstem**, a control plane that maintains identity and policy while dynamically orchestrating experts in the cortex layer.

Design goals. (1) Low–latency detection of semantic drift; (2) safe, budget–aware switching without oscillation; (3) persona consistency independent of domain expert; (4) cross–expert memory that avoids leakage; (5) measurable improvements under realistic multi–domain tasks.

2 Contributions

1. **Architecture.** A Brainstem–Cortex design: persona/safety/memory in the brainstem; domain/tool experts in the cortex.
2. **Algorithms.** (a) MGG: token/phrase/turn–level gating with hysteresis; (b) OFES: online, drift–aware routing with switching costs and risk/cost budgets.
3. **State model.** A cross–expert SWM (Session Working Memory) that carries goals, facts, tone vectors, and constraints; experts exchange summaries rather than raw traces.

4. **Evaluation suite.** MIDAS (Mid–Dialog Adaptive Switching) tasks and metrics (Switch Latency, Expert Appropriateness, Persona Consistency, Coherence Drop, Task Success, Safety Noncompliance, Switch Thrashing Index).

5. **Safety & governance.** Cross–expert guardrails, disagreement resolution, and anti–oscillation controls.

3 System Overview

3.1 Components

Brainstem (Control Plane). Router Controller (OFES + MGG); Persona Styler (PS); Safety Guard (SG); SWM; Telemetry/Bandits (TB).

Cortex (Expert Pool). Domain experts (SSA, software engineering, math, etc.), tool experts (code execution, retrieval, SQL), and formatting experts, all invoked under PS/SG.

3.2 Dataflow (ASCII, pdflatex–safe)

```
User turn -> Brainstem: SWM ingest + Drift Detector
             |
             +-> Router Controller (OFES + MGG) computes route plan
             |     (biases token-level MoE + sets phrase windows)
             |
             +-> Experts generate fragments + return deltas (facts, risks)
             |
             +-> Safety Guard enforces policy; Persona Styler harmonizes tone
             |
             +-> Compose answer + update SWM + emit telemetry
```

Figure 1: Conversation-level control with mid-utterance expert switching.

4 Problem Formulation

Let an input sequence $x_{1:T}$ produce output $y_{1:K}$ using experts $\mathcal{E} = \{e_1, \dots, e_m\}$ and a routing policy π that may change within the emission of y. We maximize utility subject to budgets and constraints:

$$\max_{\pi} \; \mathbb{E}[U(y, \mathrm{SWM}_t) - \lambda_c \mathrm{Cost}(\pi) - \lambda_r \mathrm{Risk}(\pi)] \quad \text{s.t.} \quad \mathcal{I}_p, \mathcal{I}_s, \text{ and hysteresis penalty } \gamma S. \tag{1}$$

Here $\mathcal{I}_p$ encodes persona invariants, $\mathcal{I}_s$ safety policy, and S the number of expert switches.

5 Algorithms

5.1 Multi–Granular Gating (MGG)

We extend token–level gating $g_t(e)$ with:

- **Phrase windows.** Boundaries $B = \{b_i\}$ set by drift detectors; within a window w, add bias vector β_w to expert logits.

- **Turn priors.** Initialize expert priors using SWM goals and success statistics; update with bandit feedback.
- **Hysteresis.** Switching $e_a \to e_b$ incurs cost $h(e_a, e_b)$; require margin $\Delta \geq \tau + h$ to change experts.

5.2 On–the–Fly Expert Switching (OFES)

Algorithm 1 On–the–Fly Expert Switching (OFES)

Require: tokens $x_{1:T}$, partial output $y_{1:k}$, SWM, experts $\mathcal{E}$, budgets $\mathcal{B}$

$\hat{e} \leftarrow$ prior expert; $W \leftarrow$ initial window
for each decode step k **do**
  $\phi_k \leftarrow \{\text{drift\_score}, \text{event\_triggers}, \text{risk\_est}, \text{cost\_left}, \text{perf\_signals}\}$
  if `boundary_detected`(ϕ_k) **or** `event_triggered`(ϕ_k) **then**
    $C \leftarrow$ `candidate_experts`(ϕ_k, SWM)
    for $e \in C$ **do**
      score$[e] \leftarrow$ `util_predict`(e, ϕ_k, SWM) $- \lambda_c \text{cost}(e) - \lambda_h H(\hat{e}, e) - \lambda_r \text{risk}(e)$
    $\hat{e}^\star \leftarrow \arg\max_e \text{score}[e]$
    if score$[\hat{e}^\star] -$ score$[\hat{e}] \geq \tau$ **then**
      $\hat{e} \leftarrow \hat{e}^\star$; start new window W
  $(y_k, \Delta) \leftarrow$ `decode_with`($\hat{e}, \beta_W$) ▷ β_W: phrase bias
  SWM $\leftarrow$ `integrate`(Δ) ▷ facts, citations, safety flags
  `enforce_persona_and_safety`($y_{1:k}$)
return $y_{1:K}$

Drift signals. Topic embedding deltas, code/markup events (e.g., ` ``` `, `class`, braces), math density, legal/medical NER, retrieval cues, and user directives.

6 Session Working Memory (SWM)

6.1 Principles

Typed, minimal, and leak–aware. Experts maintain private scratchpads; only summaries cross boundaries with confidence metadata.

6.2 Schema (simplified)

7 Safety and Governance

Fragment checks before composition; risky fragments are edited, re–routed, or refused with rationale. Thrashing prevention via hysteresis cost, minimum window sizes, and switch cool–downs. Disagreements escalate to retrieval expert with citations and hedged summary. SWM stores summaries/IDs, not raw traces; visibility lists and PII redaction enforced.

8 Implementation Notes

Router footprint: lightweight controller (1–3B params) for drift detection and utility prediction; <20 ms latency with batching. Bias injection: apply β_w as gating–logit offsets or adapter masks.

```
{
  "user_profile": {"name":"", "preferences":{}, "capabilities":{}},
  "tone_vector": {"formality":0.6, "warmth":0.8, "jargon":0.4},
  "goals": [{"id":"g1","text":"", "status":"open","owner":"brainstem"}],
  "facts": [{"id":"f1","text":"", "source":"retrieval|user|expert","confidence":0.86}],
  "open_threads": [{"topic":"SSA eligibility","depends":["f1","f3"]}],
  "constraints": {"safety":{"jurisdiction":"US","policy_version":"..."},
                 "persona":{"banned_phrases":[]}},
  "tool_state": {"python":{"session_id":"...","vars":{}},"retrieval":{}},
  "routing_stats": {"last_expert":"ssa","switches_this_turn":1}
}
```

Figure 2: Cross–expert SWM summary (private tool logs remain siloed).

Budgets: online bandit tunes λ_c by load/SLA; degrade gracefully to single–expert.

9 Evaluation

9.1 MIDAS Tasks

- Mid–turn pivot: SSA policy Q -> code snippet -> validator + tests.
- Multi–domain braid: math derivation -> legal compliance note -> copy polish.
- Tool pivot: NL -> SQL -> runtime fix -> executive summary.
- Safety pivot: benign question drifting into restricted domain; verify correct refusal + alternatives.

9.2 Metrics

Metric	Definition
Switch Latency (ms)	Lag from drift boundary to correct expert activation
Expert Appropriateness (P/R)	Right expert at right time vs. gold labels
Persona Consistency	Cosine similarity to pinned persona tone vector
Coherence Drop	Embedding/perplexity discontinuity across switches
Task Success	EM / pass@k / execution / citation accuracy
Safety Noncompliance	Violations per 1k turns in adversarial suites
Switch Thrashing Index	Switches per 1k tokens (lower is better)

Table 1: Core evaluation metrics for mid–conversation switching.

9.3 Baselines and Ablations

Baselines: single–expert per turn; tool routing without mid–turn switching; token–level MoE only; multi–agent handoff at turn boundaries. Ablations: remove hysteresis; remove persona styler; remove SWM summaries; static vs. adaptive windows; fixed vs. dynamic budgets.

10 Theoretical Sketch

Routing as a constrained contextual bandit with switching costs and bounded drift. With windowed decisions and confidence margins, pick–the–leader with hysteresis yields

$$\mathrm{Regret}_T + \gamma S_T = \tilde{O}\left(\sqrt{T}\right), \tag{2}$$

where S_T is the number of switches up to T and γ penalizes switching. Proof sketch in Appendix **??**.

11 Related Work

Token–level sparse MoE; instruction routing and tool use; persona/style control in dialogue; safety filtering and policy enforcement. (Citations to be added in camera–ready.)

12 Limitations

Expert calibration drift can degrade routing; rapid multi–topic pivots may incur micro–incoherence; conservative vetoes can suppress valid experts without careful tuning.

13 Broader Impacts

Potential for smoother multi–domain assistance, improved safety via vetted experts, and reduced user cognitive load. Risks include opacity of automated routing and misrouted sensitive topics; we recommend route logs and user–visible "why this expert" explanations.

A Proof Sketch: Switching–Cost Bandit

Assume utilities are Lipschitz in features with bounded variation due to drift, and boundary frequency is sublinear in token length. Using windowed updates, confidence–bound selection with a switching penalty γ yields sublinear regret while limiting S_T; aggregating gives the combined $\tilde{O}(\sqrt{T})$ bound. Full proof deferred.

B Routing Event Specification

Each event is `{type, span, features, confidence, suggested_experts}` with types: `CODE_BLOCK_START`, `MATH_HEAVY`, `LEGAL_CITATION`, `API_SIGNATURE`, `RETRIEVAL_REQUIRED`, `USER_DIRECTIVE`, `SAFETY_FLAG`. Router API: `propose_route(events, swm)`, `report(delta)`, `veto(reason)`.

C ASCII Architecture Diagram (pdflatex–safe)

```
+-------------------- Brainstem (Control Plane) --------------------+
| Drift Detector -> Router Controller (OFES + MGG)                  |
| Persona Styler -> Safety Guard (SG) -> Budgets/Telemetry          |
+-------------------------------------------------------------------+
                              | bias/plan
                              v
+------------------------- Cortex (Experts) -------------------------+
| DX: SSA   DX: Dev   DX: Math   TX: Tools                          |
+-------------------------------------------------------------------+
           fragments ^                         |
                     |                         v
            Compose output <- Persona/Safety filter -> Final answer
```

Figure 3: Architecture using ASCII only.