Recall Isn't Enough: Bounding Commitments in Personalized Language Systems

Anonymous ACL submission

Abstract

Long-context and memory systems usually treat personalization as a recall problem. In practice, many failures occur later, when a system commits: it turns noisy hints into hard constraints, drops rare witnesses, forgets downstream obligations, or answers despite infeasibility. We introduce *Contract-Bounded Evidence Activation* (CBEA) with *Lexicographic Commitment Validation* (LCV). CBEA activates a bounded evidence set using typed coverage, tail witnesses, and consequence debt; LCV validates structured commitments before prose and routes infeasible states to repair, abstention, or recontract. Across 360 fixtures and three generation backends, CBEA+LCV reaches zero failures within validator scope at 0.49–0.60 availability over attempted runs. Raw and long-context baselines with the same LCV gate reach zero only at 0.003–0.092. A shadow oracle diagnostic marks the limit: CBEA+LCV recalls 0.012 of uncompiled visible facts, while raw recalls 0.53. The result is a bounded operating point: explicit commitment control and 74–75% lower median input payload, not universal memory dominance.

1 Introduction

Long-horizon personalization must preserve confirmed user constraints while adapting to noisy, evolving state. Fluent responses can still drop hard constraints, overweight recent context, lose rare witnesses, or continue when no response is compatible with the contract. We call these *runtime control failures*: errors that occur after evidence is collected but before a commitment is safely realized. The standard alternative—stuffing the raw user history into the prompt—asks the model to rediscover which facts matter at every turn. The same sentence may be reinterpreted across turns, hard constraints flatten into soft preferences, and rare evidence disappears unless the runtime keeps an explicit obligation model.

This failure mode is distinct from ordinary memory failure. A system may retrieve the relevant sentence and still make the wrong commitment if it treats a soft hint as a hard constraint, ignores a required witness, or fails to recognize that the feasible set is empty. Conversely, a system may be fluent and useful while remaining outside any claim we can verify. The central question is therefore not only what context to show a model, but what commitments the runtime is allowed to realize after that context is selected.

We address this with runtime control bounded by contracts: runtime decisions are constrained by explicitly confirmed structured commitments covered by validators. Clarification targets missing information with high downstream impact; compilation converts confirmed information into hard predicates, evidence units, mutable state, and required coverage; CBEA selects bounded evidence; LCV filters structured commitments before prose; and infeasible states route to repair, abstention, or recontract rather than to new commitments.

The paper deliberately separates two boundaries. Inside the compiled contract, the runtime can validate structured commitments against explicit predicates and coverage requirements. Outside that boundary, uncompiled facts remain ordinary context: the method can miss them, and we measure that exclusion rather than hiding it. This is why the evaluation reports both validator-covered control failures and a shadow-oracle diagnostic over visible facts that were not compiled into the contract.

The paper makes four contributions:

1. CBEA, a budgeted evidence activation objective over local relevance, contract coverage, tail witness retention, consequence debt, and over-personalization penalties;
2. LCV, a lexicographic validation rule whose invariant is attached to structured commitments covered by validators rather than unrestricted

prose;

3. a matched evaluation across nine variants and 360 fixtures, covering hard constraints, coverage, witnesses, infeasible continuation, consequences, and repair, plus a long history payload diagnostic built from prompts selected by CBEA;
4. an audit of 90 cases by model judges over six fidelity dimensions, combined with privacy protected production aggregates as diagnostic evidence rather than causal outcome evidence.

2 Related Work and Positioning

Long horizon. Recent benchmarks already study long-horizon personalization and contextual preference inference. HorizonBench tracks evolving preferences over long simulated histories (Li et al., 2026b); CUPID evaluates contextualized alignment from interaction histories (Kim et al., 2025). Our target is not preference inference alone, but runtime control after evidence exists.

Memory. Memory systems address recall under context pressure. TiMem proposes temporal hierarchical memory consolidation (Li et al., 2026a), while OP-Bench shows that memory can be misused through irrelevance, repetition, and sycophancy (Hu et al., 2026). We also include a selector-level diagnostic against a classic MMR selector (Carbonell and Goldstein, 1998); unlike CBEA, MMR has no typed coverage, tail witness, or consequence debt terms. We therefore treat evidence activation as a constrained runtime decision before validation and realization.

Clarification. Prior work studies clarification for preference elicitation (Montazeralghaem et al., 2025; Zhang et al., 2024), profile compression (Shi et al., 2025; Qiu et al., 2025), and long context limits (Du et al., 2025). These motivate moving runtime truth from raw profile text to explicit artifacts.

Prompt compression and RAG. Compression and retrieval baselines reduce context cost, but they do not by themselves define which constraints are hard, which evidence is mandatory, or what the system must do when no feasible answer exists. Our comparison therefore includes raw history, summary, dense retrieval, long-context prompting, and tool/memory baselines, but the main claim concerns the control interface placed after evidence selection: a structured commitment is either covered by validators or outside the guarantee.

Verification. Contract-grounded planning separates retrieval, typed constraints, verification, repair, and abstention (Agarwal, 2026). Our setting differs: constraints are confirmed by the user, state evolves, evidence is incomplete, and validated commitments are realized through language. We borrow the boundary that safety claims apply only to structured commitments covered by validators.

Position. Our closest comparison is not personalization memory alone, but systems that separate retrieval, constraint representation, validation, and repair. Unlike previous work, our contribution is their integration into a runtime control method for noisy long-horizon personalized language systems.

3 Formal Runtime Model

The paper's central object is a runtime over structured commitments. Let $x_{0:t}$ be a sequence of noisy user observations, including forms, labels, free text, and previous turns. The runtime maintains a confirmed hard contract h_t, mutable state u_t, a shared evidence pool E_t, a required-coverage set R_t, and a structured action space $\mathcal{A}$.

Definition 1 (Confirmed hard contract)**.** *A hard contract is a set*

$$h_t = \{(\phi_j, \pi_j)\}_{j=1}^{m_t},$$

where each $\phi_j : \mathcal{A} \to \{0,1\}$ is a machine-checkable hard predicate and π_j is a provenance record. Only explicitly confirmed predicates belong to h_t. Inferred or unresolved constraints remain outside the immutable contract.

Definition 2 (Evidence pool)**.** *The evidence pool is a finite set*

$$E_t = \{e_i = (z_i, d_i, \rho_i, \tau_i, \kappa_i)\}_{i=1}^{n_t},$$

where z_i is an evidence unit, d_i its dimension, ρ_i provenance metadata, τ_i a tail witness indicator, and κ_i an activation cost. Selected evidence units, not raw profile text, are the operational interface to generation.

Definition 3 (Required coverage set)**.** *The runtime derives a set*

$$R_t = \Gamma(h_t, u_t, c_t)$$

of required evidence dimensions for the current turn or result. Requirements may come from confirmed hard predicates, required evidence fields,

consequence debt, local scene obligations, or no-feasible checks. Let $M_t \in \{0,1\}^{|E_t|\times|R_t|}$ be an evidence requirement coverage matrix, where $M_{ir} = 1$ means evidence unit e_i covers requirement r.

CBEA. CBEA selects evidence through a budgeted objective rather than top-k retrieval. For a candidate evidence subset $Z \subseteq E_t$, define

$$\begin{aligned} J_t(Z) = \lambda_r \operatorname{Rel}(Z, c_t) + \lambda_c \operatorname{Cov}(Z, R_t) \\ + \lambda_w \operatorname{Tail}(Z) + \lambda_d \operatorname{Debt}(Z, u_t) \\ - \lambda_o \operatorname{Over}(Z, c_t), \end{aligned}$$

where Rel measures local relevance, Cov measures requirement coverage, Tail rewards retention of rare but decisive tail witnesses, Debt rewards evidence needed for downstream obligations, and Over penalizes irrelevant or intrusive personalization. Requirement coverage is computed from M_t using a short coverage indicator:

$$\begin{aligned} \operatorname{Cov}(Z, R_t) &= \textstyle\sum_{r\in R_t} w_r\, \eta_r(Z), \\ \eta_r(Z) &= \mathbf{1}[\exists e_i \in Z : M_{ir} = 1]. \end{aligned}$$

CBEA chooses

$$Z_t^\star \in \arg\max_{Z\subseteq E_t} J_t(Z) \quad \text{s.t.} \quad \sum_{e_i\in Z} \kappa_i \le \mathcal{B}_t.$$

In implementation, this objective can be solved approximately with greedy budgeted coverage. A reserved budget $\mathcal{B}_t = \mathcal{B}_t^{main} + \mathcal{B}_t^{tail}$ lets the runtime protect required tail witnesses separately from ordinary evidence with high relevance. The corresponding ablations remove coverage, tail reservation, or consequence debt terms.

LCV. A candidate generator constructs a finite set $A_t \subseteq \mathcal{A}$ of structured commitments from $Z_t^\star$. For each candidate, LCV computes a violation vector

$$\begin{aligned} \nu_t(a) = \big(&\nu_h(a, h_t), \nu_c(a, Z_t^\star, R_t), \\ &\nu_0(a), -S(a; u_t, Z_t^\star, c_t)\big), \end{aligned}$$

where ν_h counts hard predicate failures covered by validators, ν_c counts missing required evidence coverage, ν_0 indicates commitment emission when the runtime diagnoses no feasible candidate covered by validators, and S is a soft utility score over mutable state and local context. LCV ranks candidates by lexicographic minimization:

$$\tilde{a}_t \in \arg\operatorname{lexmin}_{a\in A_t} \nu_t(a).$$

For readability, write $\bar{\nu}_{h,t}(a) = \nu_h(a, h_t)$ and $\bar{\nu}_{c,t}(a) = \nu_c(a, Z_t^\star, R_t)$. The feasible set is

$$\begin{aligned} \mathcal{F}_t = \{a \in A_t : \bar{\nu}_{h,t}(a) = 0, \\ \bar{\nu}_{c,t}(a) = 0\}. \end{aligned}$$

If $\mathcal{F}_t \neq \varnothing$, the runtime emits

$$a_t^\star \in \arg\max_{a\in\mathcal{F}_t} S(a; u_t, Z_t^\star, c_t),$$

realizes it and updates mutable state:

$$y_t = G_\psi(a_t^\star, u_t, Z_t^\star, c_t),\ u_{t+1} = U(u_t, a_t^\star, x_{t+1}).$$

If $\mathcal{F}_t = \varnothing$, the runtime must not emit a commitment as if it were feasible. LCV diagnoses an infeasibility reason such as missing evidence, contract conflict, unsupported commitment, or validator failure. Let δ_t denote this reason. The runtime calls a restricted repair operator

$$o_t = \Omega(\delta_t, h_t, u_t, Z_t^\star, c_t),$$

where o_t is repair, abstention, fallback, or explicit recontract. In our implementation, missing evidence maps to clarification, contract conflict to recontract, and unsupported or validator failing candidates to abstention or fallback. These acts do not carry commitments and are scored separately from valid structured commitments. Appendix F gives an illustrative LCV repair routing sketch.

Assumption 1 (Structured emission covered by validators)**.** *The validator exactly covers the confirmed predicates in h_t for parseable structured commitments. Repair, abstention, and recontract acts do not carry commitments: they may ask for missing information, decline to continue, or request explicit contract correction, but they must not smuggle a new commitment as advice. Timeout, no output, parse failure, partial output, and blank output are not counted as safe emissions; they are system level failures or unevaluable states with explicit denominators.*

Property 1 (Emission boundary covered by validators)**.** *Under exact validator coverage for predicates in h_t, any emitted structured commitment $a_t^\star$ satisfies all confirmed hard predicates in h_t. When $\mathcal{F}_t = \varnothing$, no structured commitment is emitted; the runtime is restricted to repair, abstention, or explicit recontract. This invariant is distinct from the oracle-level violation metrics used in the benchmark.*

Object	Runtime role	Failure measured
h_t	Confirmed hard contract	Hard constraint drift or false hardening
E_t	Shared evidence pool	Witness loss and coverage failure
u_t	Mutable turn state	Trajectory and consequence discontinuity
R_t	Required coverage set	Missing required dimensions
M_t	Evidence requirement matrix	Coverage accounting error
Z_t	Activated evidence	Prompt budget omission and tail loss
a_t	Structured commitment	Invalid feasible set membership
LCV	Lexicographic validator	Contract violation covered by validators
Ω	Repair/recontract operator	Invalid infeasible state handling
G_ψ	Surface realizer	Prose realization error

Table 1: The paper is organized around runtime objects. Evaluation metrics are derived from these objects rather than from generic fluency or engagement.

This property is a claim boundary, not a theorem over all natural language implications. The guarantee is attached to $a_t^\star$, and the surface realizer G_ψ remains a separately evaluated layer because prose can introduce unsupported implications even when the underlying commitment is valid.

4 CBEA and LCV Runtime Algorithm

Figure 1 shows the method. The model is not asked to rediscover the user contract from raw background text on every turn; the runtime compiles constraints, activates evidence, validates commitments, and only then realizes language.

Compilation. Clarification is bounded information acquisition, not open interviewing. Compilation converts confirmed answers into h_t, extracts evidence units into E_t, initializes u_t, derives R_t, and populates M_t, keeping inferred soft preferences outside the hard contract.

Activation. Evidence activation is constrained selection, not unconditional memory retrieval. CBEA adds coverage of R_t, tail witness reservation, consequence debt evidence, and overpersonalization penalties to local relevance. Turn activation prioritizes latency; result activation prioritizes synthesis and consequence preservation.

Validation. The runtime validates structured commitments before language. Hard predicates are lexicographically prior to evidence coverage, and both precede soft utility. More evidence cannot compensate for violating an explicitly confirmed boundary. If every candidate violates h_t or lacks required coverage, the runtime forbids commitments that claim feasibility and routes to $\Omega(\delta_t)$.

Realization. A valid commitment can still be phrased ambiguously or with unsupported implications, so surface realization remains a language generation problem evaluated separately (§7).

Implementation. CBEA is not a learned preference model. In our experiments, R_t and M_t are constructed from fixture schemas and runtime contract rules: confirmed hard predicates induce mandatory requirements, fixture witnesses mark covered requirements, and consequence debt obligations follow from confirmed state transitions. Oracle labels $(h^\star, E^\star, \mathcal{F}^\star)$ are used only for scoring. They are not inputs to clarification, contract compilation, CBEA selection, candidate generation, LCV validation, repair routing, or prose realization. CBEA weights are fixed design priors $(\lambda_r, \lambda_c, \lambda_w, \lambda_d, \lambda_o) = (1, 2, 2, 1, 1)$, not learned, tuned on a development set, or tuned after the fact for metric maximization; they encode the priority that coverage and tail witnesses dominate ordinary relevance. Candidates share a schema (commitment type, evidence ids, claimed predicates, required slots, repair status, surface text); validators cover hard contract consistency, required evidence coverage, infeasible emission, and repair reason validity. These choices make the method auditable but also bound the claim: results test this fixed instantiation, not an optimized learned selector. Appendix B lists the reporting schema.

5 Evaluation and Benchmark Protocol

The evaluation follows the formal objects in Table 1. We use narrative decision support as a hostile stress test because it combines noisy self report, delayed consequences, household or social spillovers, long free text background, and recovery after drift. The benchmark does not claim to measure real world decision quality.

5.1 Fixture Construction

Fixtures derived from real users are synthetic or composite cases derived from aggregate failure patterns, never raw histories. The key transformation

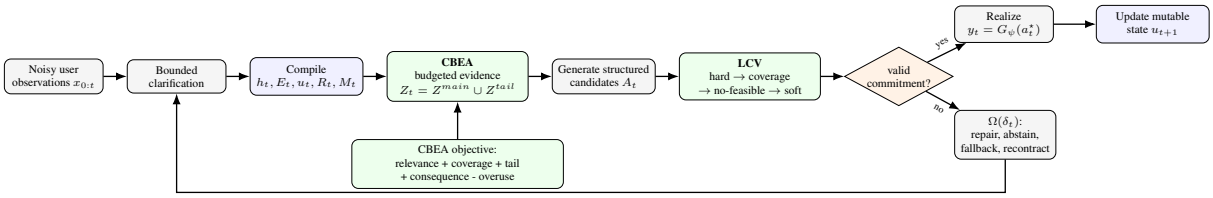

Figure 1: Runtime control with validator gates, CBEA, and LCV. Evidence is activated under budget, commitments are validated before prose realization, and infeasible states route to repair, abstention, fallback, or recontract.

is from observed failure families to controlled fixtures with known evaluator labels: confirmed predicates $h^\star$, evidence witnesses $E^\star$, expected feasible set $\mathcal{F}^\star$, and allowed repair or abstention behavior. The construction pipeline—aggregate failure patterns → abstract stress mechanism → synthetic/composite profile rewrite → structured labels → controlled offline fixture—is detailed in Appendix C, which also reports the exported fixture buckets and stress surface mapping.

The fixtures cover seven stress surfaces: false hardening, hidden exception, witness drop, infeasible continuation, consequence debt, overpersonalization, and surface mismatch. Six are exported as primary fixture buckets in the matched manifest; overpersonalization is retained as an activation penalty rather than a separate bucket (Appendix C).

5.2 Metrics

Let $\mathcal{D}$ be a controlled benchmark set, $\mathcal{D}_{emit}$ the subset with parseable emitted structured commitments, and $\mathcal{D}_0$ the subset whose oracle feasible set is empty. Let $q_f = 1$ when fixture f emits a parseable structured commitment and $q_f = 0$ otherwise. Structured output availability is

$$\text{Avail} = \frac{1}{|\mathcal{D}|} \sum_{f \in \mathcal{D}} q_f.$$

Let $v_f = 1$ when $\hat{a}_f$ violates at least one oracle predicate in $h_f^\star$, and $v_f = 0$ otherwise. Oracle hard constraint violation rate is reported over parseable emitted commitments:

$$\text{OHCVR} = \frac{1}{|\mathcal{D}_{emit}|} \sum_{f \in \mathcal{D}_{emit}} v_f.$$

This oracle level metric is stricter than the invariant covered by validators above: it can count hidden, uncompiled, or incorrectly realized fixture predicates that were not certified by V_h. Let $n_f = 1$ when the emitted control act carries a commitment in an infeasible fixture, and $n_f = 0$ for repair, abstention, or recontract. The infeasible emission rate is

$$\text{NFER} = \frac{1}{|\mathcal{D}_0|} \sum_{f \in \mathcal{D}_0} n_f,$$

where a commitment output in this setting is an infeasible emission failure. Evidence coverage failure and consequence continuity failure are computed analogously from fixture labels. Repair correctness is computed over $\mathcal{D}_{repair}$, the subset of fixtures whose evaluator label expects repair, abstention, or recontract behavior (including but not limited to $\mathcal{D}_0$); it scores 1 when the emitted control act matches the expected repair category without making a commitment. Surface realization is evaluated separately in the blinded model judge audit, where judges score surface coherence from rubric labels rather than an automatic proxy. The surface mismatch fixture bucket still contributes its structured contract, coverage, witness, consequence, infeasible emission, and repair labels to the automatic metrics in Table 2; only surface coherence itself is not scored by an automatic proxy.

Metrics requiring a structured commitment are reported on $\mathcal{D}_{emit}$ with denominator shown. Aggregate system pass rates include timeout, no output, parse failure, partial output, blank output, and unrepaired error states as failures rather than silently removing them (Appendix C).

5.3 Baselines and Ablations

The run includes two isolating variants: validator only separates validation and repair from evidence activation, and runtime without CBEA keeps structured state and repair while replacing CBEA with evidence selection without tail reserve. We use these rows as targeted component diagnostics for the activation and validation interfaces. Appendix B, Table 5 summarizes the nine comparison interfaces.

Model reporting. Controlled comparisons report model identifiers, decoding settings, context/generation budgets, retrieval budgets, tool availability, and memory availability. Production

traffic is not used for model comparison because deployed provider/model mix changes over time and is only operational context.

Reproducibility. We provide an anonymized artifact that preserves privacy and includes the fixture generator, 360-fixture benchmark, runtime harness with CBEA selector and LCV validators, released model output result tables, metric summarizers, paired and case cluster bootstrap scripts, selector level MMR diagnostic, backend sensitivity configs, and privacy boundary checks. It supports reproduction of Table 2, the long history payload diagnostic, targeted ablations, backend sensitivity operating points, and model judge bootstrap summaries; it contains no raw user histories or production text, and production evidence is released only as aggregate counts.

6 Controlled Offline Experiments

We ran 360 synthetic and composite fixtures across the nine variants (Appendix B, Table 5). All non-oracle variants used MiniMax-M2.7 in a matched offline run with temperature 0.2, a 2200-token cap, up to three JSON parse retries, and a 180-second timeout. The final run contains 3,240 attempted and evaluable rows. Oracle evidence is deterministic and included only as a reference point.

Method	Att.	Inv.	Struct.	Repair	Cost
Raw prompt stuffing	360	0	**0.7833**	0.4571	3218
Summarized profile	360	0	0.4500	0.6786	3395
Dense retrieval RAG	360	0	0.7333	0.3571	2612
Long-context LLM	360	0	0.7278	0.4786	3279
Tool/memory agent	360	0	0.7500	0.4643	3189
Validator-only	360	0	0.4028	1.0000	**2778**
Runtime w/o CBEA	360	0	0.5000	1.0000	3542
CBEA + LCV runtime	360	0	0.5000	1.0000	3178
Oracle (ref.)	360	0	0.6111	1.0000	–

Table 2: Matched offline operating points over 360 synthetic/composite fixtures (MiniMax-M2.7). Att.: attempted runs; Inv.: invalid; Struct.: structured commitment availability; Repair: $\mathcal{D}_{repair}$ correctness; Cost: average prompt cost units. Appendix H reports OHCVR/ECF/Wit./Cons./NFER for all nine methods; Appendix L reports the uncompiled context boundary.

Table 2 and Figure 2 show a shift in operating point, not broad dominance. CBEA+LCV emits structured commitments on 0.5000 of fixtures with zero measured failures on covered surfaces; raw prompt stuffing emits on 0.7833 but has failures across every covered surface. The remaining CBEA+LCV fixtures route to repair, abstention, or recontract without making new commitments, with $\mathcal{D}_{repair}$ correctness $= 1.0000$. Raw remains competitive on availability because the fixtures expose relevant evidence directly. The runtime contribution is to make constraint, coverage, witness, and consequence behavior *measurable and enforceable* on emitted commitments.

This operating point is intentionally conservative. A system that emits on more cases can look better under a simple availability metric while silently violating hard predicates or losing required evidence. A system that refuses too often can also drive failure rates to zero without solving the activation problem. We therefore read Table 2 together with Figure 2: the target is not the upper-right corner, but a point where emitted commitments remain covered and unrepaired cases are explicitly routed.

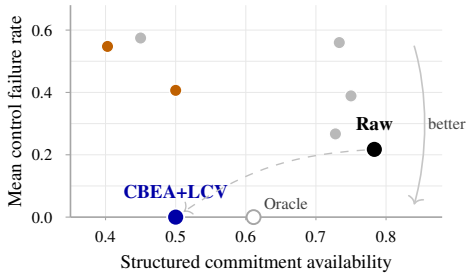

Figure 2: Operating point view of Table 2. Raw prompt stuffing emits more commitments but has higher control failures on covered surfaces; CBEA+LCV emits fewer commitments but reaches the zero failure operating point within the validator scope. Other baselines (orange: with LCV gates; gray: ungated) are shown as context, not as a universal dominance claim. Dashed arrow: design shift raw→CBEA+LCV.

Property 1 explains why LCV can suppress infeasible commitments or commitments that violate validators; it does not ensure that the generator sees the right witnesses or consequence obligations. Those surfaces depend on activation: validator-only and runtime without CBEA still lose coverage, witnesses, and consequences (Appendix H). Thus validation alone can refuse infeasible cases, but CBEA supplies the evidence state that makes validation useful.

Targeted ablations (Appendix G) isolate validation, repair, and coverage/tail activation; a selector diagnostic (Appendix I) shows CBEA retains 0.9970 of typed control evidence versus 0.6960 for MMR. Backend sensitivity and the Hy3-preview output budget diagnostic are summarized below; horizon complexity and long history payload diagnostics reuse the same harness.

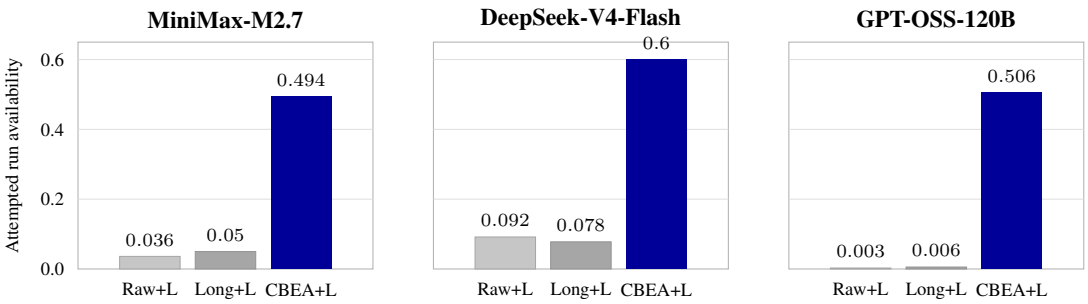

Figure 3: Backend sensitivity operating points over the matched 360-fixture comparison with LCV gates. Bars show attempted run structured commitment availability for Raw+LCV, Long-context+LCV, and CBEA+LCV; all cells have zero measured failures covered by validators on emitted commitments. Full system cost diagnostics are in Appendix J.

Backend sensitivity. Across the three backends, CBEA+LCV attempted run availability is 0.4944, 0.6000, and 0.5056; raw and long-context baselines with LCV gates reach the same zero failure level only at 0.0028–0.0917. The pattern preserves direction across two model families and three labs, but remains a sensitivity diagnostic rather than a robustness claim. Hy3-preview is reported separately: at 2200 tokens it does not emit parseable structured commitments, while an 8800-token diagnostic is small and not matched to the 360-fixture protocol.

Activation boundary. Property 1 suppresses infeasible commitments or commitments that violate validators, but it cannot make missing witnesses or consequence obligations visible. CBEA's role is therefore not a tighter validator; it is a selector whose typed coverage, tail witness, and consequence debt terms put the right evidence units into $Z_t^\star$ before validation runs.

Shadow exclusion boundary. A shadow oracle diagnostic (Appendix L) maps visible but uncompiled facts outside the compiled guarantee. CBEA+LCV is nearly inert on them; raw prompt stuffing retains more. This is not a second success criterion for CBEA+LCV, but the exclusion side of the guarantee. The diagnostic prevents the main zero-failure result from being read as universal memory coverage: CBEA+LCV controls what it compiles, and raw history exposes more uncompiled facts while failing more often on covered commitments. The contribution is a precisely characterized operating point within the validator scope; combining that point with broader uncompiled-context awareness is future work.

Production-scale diagnostics. A privacy-protected deployment export (524 attempted sessions over a 720-hour window, 447 in the semantic pool after operational failures are separated, 6,018 evaluable turns) supports deployment scale and reliability diagnostics only; it is not correctness evidence. Full denominators are in Appendix C.

7 Blinded Model Judge Fidelity Audit

We run a reproducible diagnostic audit with two open-weight instruction-tuned model judges, DeepSeek-V4-Pro and Qwen/Qwen3.6-35B-A3B, accessed through third-party inference endpoints; open-weight status identifies the model, not the serving mode. Judges see blinded user-visible outputs for 90 held-out cases and three systems, plus a plain-English runtime-control reference. They do not see method names, automatic scores, raw production data, credentials, or oracle labels. Judges score six fidelity dimensions on a 0–2 rubric and choose the most faithful output per case. We use this as model-based fidelity evidence, not decision-quality evidence; model judges have known reliability and bias limits (Liu et al., 2023; Zheng et al., 2023; Gu et al., 2024; Ye et al., 2024).

System	Overall	NFE	Surf.	Win
Raw	**1.6778**	1.7667	**1.8778**	30.6%
CBEA+LCV	1.6509	**1.7944**	1.8167	**50.0%**
Validator	1.3843	1.5278	1.7889	17.8%

Table 3: Model-judge diagnostic. Overall, NFE (infeasible handling), and Surf. are dimension means on a 0–2 rubric; Win is the share of 180 case-winner selections across two judges and 90 cases, not human preference. **Bold** marks the column max. Detailed judge scores are in Table 21; inter-judge agreement is in Table 22.

The audit is conservative by design: dimension means remain close, with raw slightly higher on

overall and surface coherence, while winner selections favor CBEA+LCV under both judges (Table 3). A case-cluster bootstrap gives CBEA+LCV a 50.0% winner share with 95% CI [41.7, 58.3] and a 19.4-point margin over raw with CI [5.0, 33.9] (Appendix M). We treat this as secondary model-based fidelity evidence, not human preference.

8 Discussion

CBEA and LCV change the operating point of the evaluated runtime. CBEA makes evidence selection an explicit budgeted optimization; LCV makes commitment feasibility a lexicographic decision rather than a soft preference. Raw prompt stuffing remains a strong baseline because the benchmark exposes relevant evidence directly in the prompt, so the contribution is narrower and systems-oriented: runtime control failures become measurable, selected witnesses are protected under bounded activation, and infeasible states route to non-commitment acts.

The main comparison is therefore not "short prompt beats long prompt." It is a trade between implicit and explicit control. Raw prompting gives the generator broad access to text but leaves the interpretation of that text to the model at each turn. CBEA+LCV narrows the active state, which loses some uncompiled context, but turns covered constraints, required evidence, witnesses, and repair states into objects that can be checked before prose. The shadow diagnostic is important precisely because it shows this cost rather than smoothing it away.

The long-history diagnostic gives the systems interpretation. With synthetic histories lengthened and runtime state held fixed, raw prompt stuffing carries the full archive while CBEA carries the selected evidence subset, reducing median input payload by 74–75% (15.3k–16.3k tokens per case) across MiniMax, DeepSeek, and Qwen endpoints. Latency varies with provider behavior and output length, so input-payload reduction is the cross-endpoint result.

The evidence design separates deployment realism from comparative proof: production aggregates show scale and failure prevalence; fixtures anchor controlled stress surfaces without exposing user data; controlled experiments establish automatic runtime effects; and the model-judge audit is secondary diagnostic evidence, not human validation. The result is a systems and evaluation contribution, not a theory of language control or evidence of real-world decision improvement.

9 Conclusion

CBEA and LCV separate evidence activation from commitment validation in a long-horizon personalization runtime. Across matched, backend-sensitivity, long-history, shadow-boundary, and model-judge diagnostics, CBEA+LCV makes runtime control failures measurable and reaches zero measured failures within validator scope at substantially higher structured-commitment availability than validation applied to raw or long-context baselines. The contribution is a runtime-control operating point: a way to trade raw context breadth for explicit commitment control, with the loss of uncompiled context measured rather than hidden.

10 Limitations

The runtime guarantee is limited to explicitly confirmed structured commitments covered by validators. It does not certify unrestricted natural language semantics. Surface realization can still introduce ambiguity or unsupported implications after a valid commitment has been selected.

Production aggregates are observational: they establish deployment scale, reliability, and failure prevalence, not causal improvement or user-outcome quality. The benchmark is diagnostic, not distributional; results should not be read as average user-traffic performance. The fixtures intentionally isolate prespecified runtime-control stress surfaces rather than estimating their frequency in production traffic. Fixtures are synthetic or composite transformations of observed failure patterns, not raw user histories or a representative user sample.

The architecture depends on the quality of clarification, contract compilation, evidence labeling, and validator coverage; missing witnesses, false hardening, and incomplete validators remain possible failure modes. The fidelity audit is model-based and should be read as a reproducible diagnostic signal rather than human validation. We do not claim novelty in preference inference, memory consolidation, overpersonalization diagnosis, or contract-grounded verification in isolation; the contribution is their integration as a validator-covered runtime-control operating point.

11 Ethical Considerations and Privacy

Long-horizon personalized systems can create an illusion of certainty, especially in decision support settings. The runtime is designed to reduce one class of failure—drift across explicitly confirmed boundaries—but it does not justify medical, legal, financial, mental health, relationship, career, or life outcome claims.

The evaluation uses only privacy protected production aggregates and synthetic, composite, or heavily redacted fixtures. We exclude raw histories, raw biographies, raw core dilemmas, emails, user IDs, session tokens, exact payment records, exact timestamps tied to individuals, and identifiable life event combinations. The judge prompts and artifacts do not include credentials, method names for blinded outputs, automatic scores, raw production histories, or user identifiers.

References

Sahil Agarwal. 2026. Contract2plan: Verified contract-grounded retrieval-augmented optimization for BOM-aware procurement and multi-echelon inventory planning. *Preprint*, arXiv:2601.06164.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336.

Yufeng Du, Minyang Tian, Srikanth Ronanki, Subendhu Rongali, Sravan Babu Bodapati, Aram Galstyan, Azton Wells, Roy Schwartz, Eliu A Huerta, and Hao Peng. 2025. Context length alone hurts LLM performance despite perfect retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 23281–23298, Suzhou, China. Association for Computational Linguistics.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2024. A survey on LLM-as-a-judge. *Preprint*, arXiv:2411.15594.

Yulin Hu, Zimo Long, Jiahe Guo, Xingyu Sui, Xing Fu, Weixiang Zhao, Yanyan Zhao, and Bing Qin. 2026. OP-Bench: Benchmarking over-personalization for memory-augmented personalized conversational agents. *Preprint*, arXiv:2601.13722.

Tae Soo Kim, Yoonjoo Lee, Yoonah Park, Jiho Kim, Young-Ho Kim, and Juho Kim. 2025. CUPID: Evaluating personalized and contextualized alignment of LLMs from interactions. *Preprint*, arXiv:2508.01674. Accepted to COLM 2025.

Kai Li, Xuanqing Yu, Ziyi Ni, Yi Zeng, Yao Xu, Zheqing Zhang, Xin Li, Jitao Sang, Xiaogang Duan, Xuelei Wang, Chengbao Liu, and Jie Tan. 2026a. TiMem: Temporal-hierarchical memory consolidation for long-horizon conversational agents. *Preprint*, arXiv:2601.02845.

Shuyue Stella Li, Bhargavi Paranjape, Kerem Oktar, Zhongyao Ma, Gelin Zhou, Lin Guan, Na Zhang, Sem Park, Lin Chen, Diyi Yang, Yulia Tsvetkov, and Asli Celikyilmaz. 2026b. Horizonbench: Long-horizon personalization with evolving preferences. *Preprint*, arXiv:2604.17283.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG evaluation using GPT-4 with better human alignment. *Preprint*, arXiv:2303.16634.

Ali Montazeralghaem, Guy Tennenholtz, Craig Boutilier, and Ofer Meshi. 2025. Asking clarifying questions for preference elicitation with large language models. *Preprint*, arXiv:2510.12015.

Yilun Qiu, Xiaoyan Zhao, Yang Zhang, Yimeng Bai, Wenjie Wang, Hong Cheng, Fuli Feng, and Tat-Seng Chua. 2025. Measuring what makes you unique: Difference-aware user modeling for enhancing LLM personalization. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 21258–21277, Vienna, Austria. Association for Computational Linguistics.

Yunxiao Shi, Wujiang Xu, Zhang Zeqi, Xing Zi, Qiang Wu, and Min Xu. 2025. PersonaX: A recommendation agent-oriented user modeling framework for long behavior sequence. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5764–5787, Vienna, Austria. Association for Computational Linguistics.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, Nitesh V. Chawla, and Xiangliang Zhang. 2024. Justice or prejudice? quantifying biases in LLM-as-a-judge. *Preprint*, arXiv:2410.02736.

Shenao Zhang, Donghan Yu, Hiteshi Sharma, Han Zhong, Zhihan Liu, Ziyi Yang, Shuohang Wang, Hany Hassan, and Zhaoran Wang. 2024. Self-exploring language models: Active preference elicitation for online alignment. *Preprint*, arXiv:2405.19332.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and chatbot arena. *Preprint*, arXiv:2306.05685.

A Reproducibility Artifact

Table 4 lists the anonymous review artifact for reproducing reported diagnostics; it excludes author/repository identifiers, credentials, raw production or user data, and exact user-linked timestamps.

Checklist item	Artifact contents
Fixtures	Generator, JSON schema, and 360 released fixtures
Results	Model-output score tables, ablations, backend sensitivity, and long-history payload diagnostics
Metrics	Summarizers, denominator checks, release-table checks, and public table cases
Uncertainty	Paired bootstrap and case-cluster bootstrap scripts
Baselines	Matched harness, LCV-gated baselines, and selector-level MMR diagnostic
Judge audit	Blinded 90-case items, annotation key, labels, and winner selections
Privacy	Aggregate-only production diagnostics and privacy-boundary checks

Table 4: Anonymous reproducibility artifact card.

B CBEA and LCV Reporting Details

Table 5 summarizes the comparison interfaces in the matched offline run. Table 6 reports the fixed method choices used by CBEA+LCV. The values are design-prior settings, not learned parameters or post-hoc tuning.

Method	Ctrl.	Interface and purpose
Raw prompt stuffing	No/No	Full profile or memory text in prompt; naive personalization under budget pressure
Summarized profile	No/No	LLM-compressed user history; product-style compression baseline
Dense retrieval RAG	No/No	Embedding top-k snippets; tests whether similarity retrieval is enough
Long-context LLM	No/No	Maximum feasible history; tests whether longer context alone solves control
Tool/memory agent	No/No	Write memory, retrieve memory, call tools; strong agent-memory baseline
Validator-only	Yes/Yes	Structured candidates filtered by V_h; no CBEA activation
Runtime without CBEA	Yes/Yes	Structured state and validator with non-tail evidence selection
CBEA + LCV runtime	Yes/Yes	$h_t, E_t, u_t, R_t, M_t, Z_t, a_t$ with Ω; main method
Oracle evidence upper bound	Yes/Yes	Oracle-selected evidence witnesses; non-deployable upper bound

Table 5: Comparison set for the matched offline run. Ctrl. is hard-contract/repair-path availability. Oracle evidence is a reference upper bound rather than a deployable baseline.

Component	Reported implementation
CBEA weights	$(\lambda_r, \lambda_c, \lambda_w, \lambda_d, \lambda_o) = (1, 2, 2, 1, 1)$ for relevance, coverage, tail witness, consequence debt, and overuse penalty
Budget policy	Greedy budgeted coverage with tail-witness reservation
Requirement sources	Confirmed hard predicates; required dimensions and slots; runtime-detected missing requirements; tail-witness requirements; consequence obligations compiled from confirmed state; no-feasible guards derived by runtime rules
Coverage matrix	Binary evidence-unit by requirement matrix; $M_{ir} = 1$ when evidence unit e_i satisfies requirement r
Candidate schema	Selected option; commitment type; hard predicates used; evidence witness ids; covered requirements; consequence obligations; repair or abstain reason; surface requirements
Validator predicates	Hard-contract consistency; required-evidence coverage; no-feasible emission guard; repair reason validity; surface-requirement presence
LCV ordering	Hard predicate violations, then coverage violations, then no-feasible emission, then soft score

Table 6: Fixed CBEA/LCV reporting details for the matched offline run.

C Production Aggregate Details

Table 7 gives the production data-wash denominators used for deployment-scale diagnostics.

Surface	Count	Rate	Use
Attempted sessions	524	100.0%	Deployment denominator
Semantic session pool	447	85.3%	After data wash
Completed sessions	398	76.0%	Funnel state
Final evaluable sessions	323	61.6%	Result analysis
Active/unfinished	98	18.7%	Not correct outputs
Operational invalid	77	14.7%	Excluded from semantics
Evaluable turns	6,018	99.0%	Turn denominator
Provider requests	28,815	100.0%	Request denominator
Provider successes	19,988	69.4%	Reliability

Table 7: Privacy-safe production data-wash export. These values support deployment diagnostics only.

Runtime-control fields were populated for most semantic sessions: runtime context for 445 of 447, contract-state markers for 396, required dimensions for 391, selected dimensions for 396, and consequence debt for 323.

Denominator discipline. Evaluation scripts use the same denominator discipline as reporting. Production sessions define deployment scale, task mix, completion, error rate, provider reliability, latency, and runtime activation; completed/evaluable outputs are used only for readiness summaries. Attempted benchmark runs define pass, timeout, no-output, and no-feasible handling rates; benchmark outputs with text define fidelity labels and deterministic proxy validation. Unfinished, errored, partial, blank, provider-side, and parsing-failure states are reliability outcomes, not correct semantic outputs.

Fixture and data boundaries. The controlled fixtures are synthetic or composite and preserve seven stress surfaces. The 360-fixture manifest exports six primary buckets with 60 fixtures each (Table 8); over-personalization is evaluated as an activation penalty and diagnostic rather than a separate bucket. Fixtures exclude raw production histories, biographies, session identifiers, payment records, exact user-linked timestamps, and identifiable life-event combinations.

Bucket	N	Stress surface
falsehard	60	False hardening
exception	60	Hidden exception
tail	60	Witness drop
infeasible	60	No feasible candidate
debt	60	Consequence debt
surface	60	Surface mismatch

Table 8: Primary exported fixture buckets in the matched 360-fixture run. Counts are over unique fixture identifiers in the merged evaluation manifest; each fixture is evaluated under all nine variants.

Stress surface	Primary metric and test
False hardening	False-hardening rate; whether noisy hints become hard constraints
Hidden exception	OHCVR; whether validation respects exception scope
Witness drop	Evidence coverage failure; whether activation preserves rare evidence
No feasible candidate	NFER and repair correctness; whether the system emits or repairs/abstains
Consequence debt	Consequence continuity failure; whether state carries downstream obligations
Over-personalization	Personalization diagnostic; whether irrelevant memory is suppressed
Surface mismatch	Surface coherence audit; whether structure survives prose realization

Table 9: Benchmark stress surfaces. Each fixture has structured labels, so failure metrics are defined over runtime objects rather than generic one-turn quality.

D Long-History Prompt-Payload Diagnostic

Table 10 reports a separate 50-fixture payload diagnostic. Only the user-visible history archive is lengthened: confirmed contracts, compiled requirements, and runtime rules are unchanged. Raw receives the full history; CBEA+LCV receives the evidence subset selected by the harness. The runner records provider usage and wall-clock time, but does not parse outputs or perform JSON retries.

Endpoint	Raw in p50	CBEA in p50	Δ in p50	Raw out p50	CBEA out p50	Δ lat mean	Δ lat p50
MiniMax	20,401	5,154	15,258	700	700	+1.74s	+1.54s
DeepSeek	21,364	5,484	15,929	700	700	+2.06s	+1.97s
Qwen	21,709	5,376	16,279	2,194	2,657	-0.80s	-1.33s

Table 10: Long-history prompt-payload diagnostic. Inputs and outputs are provider-reported tokens. Δ is raw prompt stuffing minus CBEA+LCV; positive latency means CBEA was faster within that endpoint. The supported cross-endpoint result is input payload reduction, not latency.

Median input falls by 74–75% (15.3k–16.3k tokens per case). Mean latency intervals are favorable on MiniMax (+1.74s, 95% CI [1.18, 2.29]) and DeepSeek (+2.06s, [1.45, 2.69]); Qwen keeps the same input reduction but has longer CBEA completions and a mean latency interval crossing zero (-0.80s, [-2.01, 0.42]). We therefore treat input payload reduction as the supported cross-endpoint result and latency as endpoint/output-length dependent.

Surface	Diff.	95% CI
Structured availability	-0.2833	$[-0.3361, -0.2306]$
Parse retries per fixture	$+0.0333$	$[-0.0250, 0.0944]$
All-row repair correctness	$+0.5429$	**[0.4571,0.6286]**
Observed short-prompt latency, ms	$+1141$	$[-882, 3132]$

Table 11: Paired bootstrap intervals over 360 fixtures for CBEA+LCV minus raw prompt stuffing. Positive is better for availability and repair correctness; negative is better for retries and short-prompt latency. **Bold** marks intervals excluding zero. The latency row is separate from the long-history payload diagnostic in Table 10. Negative structured availability reflects LCV routing infeasible fixtures to non-commitment control acts rather than unsupported commitments.

E Horizon-Complexity Stability Diagnostic

The fixtures do not contain real turn-depth labels, so we do not report a turn-depth causal analysis. As a conservative proxy, Table 12 groups the matched run by the number of required domains in each fixture. The split reuses the same 360-fixture outputs and adds no new model calls. CBEA+LCV keeps oracle-level OHCVR, witness-drop, and consequence-continuity failures at zero in every group, while repair correctness remains 1.0. Parse retries are mixed and are not used as a main claim. Oracle-level OHCVR remains an end-to-end fidelity diagnostic rather than a validator-covered invariant.

Group	Method	Fix.	Struct.	OHCVR	Wit.	Cons.	Repair	Retry
2-domain	Raw	144	119	0.1176	0.0672	0.4622	0.4286	0.13
2-domain	CBEA+LCV	144	80	0.0000	0.0000	0.0000	1.0000	0.15
3-domain	Raw	144	106	0.1792	0.0943	0.4623	0.5000	0.15
3-domain	CBEA+LCV	144	61	0.0000	0.0000	0.0000	1.0000	0.24
4-domain	Raw	72	57	0.0526	0.0175	0.3860	0.4286	0.19
4-domain	CBEA+LCV	72	39	0.0000	0.0000	0.0000	1.0000	0.14

Table 12: Horizon-complexity diagnostic over the matched 360-fixture run. Groups are based on the number of required domains in each fixture, not real conversation turn depth. OHCVR, Wit., and Cons. are lower-is-better diagnostic rates; Repair is higher-is-better repair or abstention correctness; Retry is mean parse retries per fixture.

F LCV Repair Case Sketch

Table 13 gives a qualitative sketch from one relocation-career fixture. The confirmed contract includes no new unsecured debt, preserving a weekly caregiving obligation, and not asserting partner approval without a witness. Activated evidence contains the offer constraints, debt boundary, and caregiving schedule; partner backup is missing. The example is not a decision quality case study. It shows how LCV routes infeasible or unsupported states before prose realization.

LCV diagnosis	Detected condition	Repair action
Missing evidence	Candidate assumes backup care or partner approval without an activated witness.	Clarify the missing premise; no recommendation is emitted.
Contract conflict	Candidate requires protected caregiving time or new debt, contradicting h_t.	Recontract: state the conflict and ask whether to revise the boundary; do not emit the infeasible commitment.
Unsupported commitment	Candidate adds household approval or debt-safety claims not supported by evidence.	Abstain or fallback to a verified comparison with explicit uncertainty.

Table 13: Illustrative LCV repair routing for one synthetic and composite fixture. The table distinguishes clarification, recontract, and abstention or fallback as separate non-commitment control acts.

G Targeted Runtime Ablations

Table 14 reports targeted ablations from a separate 360-fixture component-diagnostic run. The comparison is diagnostic: it tests whether removing a runtime component degrades the surface that component is designed to control, rather than replacing the main matched comparison in Table 2. Absolute values should be read as targeted component diagnostics rather than as additional baseline rows for Table 2.

Variant	Inv.	Struct.	OHCVR	ECF	Wit.	Cons.	NFER	Rep.
CBEA+LCV diagnostic	0	0.8361	**0.0631**	**0.0532**	**0.0465**	0.0498	**0.0066**	**0.9508**
No validator	0	0.8583	0.1521	0.1489	0.1553	0.0971	0.0680	0.5417
No repair/abstain	0	1.0000	0.2000	0.0222	0.0250	0.0139	0.1667	0.0000
No coverage/tail	0	0.8167	0.2143	1.0000	1.0000	1.0000	0.0068	0.8657

Table 14: Targeted ablation results from a separate 360-fixture component diagnostic. Lower is better for Inv., OHCVR, ECF, Wit., Cons., and NFER; higher is better for Struct. and Rep. **Bold** marks intended control surfaces where the full diagnostic runtime improves over targeted removals. This run is not a replacement for the matched comparison in Table 2.

H Validator-Covered Control Failures: 9-Method Detail

Table 15 expands the matched nine-method comparison in Table 2. LCV suppresses infeasible commitments by construction, but validation alone does not guarantee evidence coverage, witness retention, or consequence continuity; CBEA+LCV is the only non-oracle row with zeros across all covered metrics.

Method	Struct.	OHCVR	ECF	Wit.	Cons.	NFER
Raw prompt stuffing	0.7833	0.1277	0.4113	0.0674	0.4468	0.0333
Summarized profile	0.4500	0.2469	1.0000	0.5988	0.9938	0.0333
Dense retrieval RAG	0.7333	0.1364	0.2803	1.0000	1.0000	0.3833
Long-context LLM	0.7278	0.1565	0.5344	0.0649	0.5458	0.0333
Tool/memory agent	0.7500	0.1704	0.4407	0.6926	0.6074	0.0333
Validator-only	0.4028	0.2759	0.4621	1.0000	1.0000	$0^{\dagger}$
Runtime w/o CBEA	0.5000	0.2444	0.4889	0.7778	0.5222	$0^{\dagger}$
CBEA + LCV runtime	0.5000	0	0	0	0	$0^{\dagger}$
Oracle evidence (ref.)	0.6111	0	0	0	0	0

Table 15: Validator-covered control failures on the matched 360-fixture MiniMax-M2.7 run. OHCVR: hard-constraint violation; ECF: evidence-coverage failure; Wit.: witness-drop rate; Cons.: consequence-continuity failure; NFER: no-feasible emission rate. †NFER $=$ 0 by Property 1. Validator-only and Runtime w/o CBEA show that LCV gating alone does not yield the CBEA+LCV zeros.

I Selector-Level MMR Diagnostic

Table 16 compares CBEA activation with a classic relevance-diversity MMR selector on the same 360 fixtures and 12-unit evidence budget. This selector-only diagnostic asks whether relevance-diversity alone recovers the typed control evidence that LCV later needs. MMR uses local scenario and observation tokens as the query and has no access to coverage, tail-witness, or consequence-debt terms.

Selector	Avg. $\lvert Z \rvert$	Hard	ReqW	Tail	Debt	Control
CBEA selector	10.67	**1.0000**	**1.0000**	**1.0000**	**0.9889**	**0.9970**
MMR relevance-diversity	12.00	**1.0000**	0.9444	0.6667	0.0000	0.6960

Table 16: Selector-level MMR diagnostic over the same fixture evidence universe and budget, without model generation. Columns report mean recall for hard constraints, required witnesses (ReqW), tail witnesses, consequence debt, and their union (Control).

J Backend Sensitivity and Operating Points

Table 17 reruns the 360-fixture comparison on MiniMax-M2.7, DeepSeek-V4-Flash, and GPT-OSS-120B, applying the same LCV gate to Raw, Long-ctx, and CBEA+LCV. This is a validator-scope sensitivity diagnostic, not robustness evidence or a second model-judge audit; Hy3-preview is separate because it fails to emit parseable commitments at the matched 2200-token budget.

We report structured-output cost separately from covered correctness: **Inv.** is invalid runs, **Bdg.** budget-exhausted JSON parse failure, and **Long.** output at or above the 2200-token budget. All emitted commitments have zero validator-covered failures, but availability differs: CBEA+LCV reaches 0.4944, 0.6000, and 0.5056, while LCV-gated raw and long-context baselines reach the same failure level only at 0.0028–0.0917 availability.

Backend	Method	Inv.	Bdg.	Long.	Avail.	OHCVR	ECF	Wit.	Cons.
MiniMax-M2.7	Raw + LCV	5	5	275	0.0361	0	0	0	0
MiniMax-M2.7	Long-ctx + LCV	1	1	287	0.0500	0	0	0	0
MiniMax-M2.7	CBEA + LCV	1	1	211	**0.4944**	0	0	0	0
DeepSeek-V4-Flash	Raw + LCV	0	0	0	0.0917	0	0	0	0
DeepSeek-V4-Flash	Long-ctx + LCV	0	0	0	0.0778	0	0	0	0
DeepSeek-V4-Flash	CBEA + LCV	0	0	0	**0.6000**	0	0	0	0
GPT-OSS-120B	Raw + LCV	0	0	7	0.0028	0	0	0	0
GPT-OSS-120B	Long-ctx + LCV	0	0	9	0.0056	0	0	0	0
GPT-OSS-120B	CBEA + LCV	1	0	15	**0.5056**	0	0	0	0

Table 17: Backend sensitivity over 360 fixtures $\times$ 3 methods $\times$ 3 backends (3,240 attempted rows) at a 2200-token budget. **Inv.**: invalid runs; **Bdg.**: budget-exhausted JSON parse failures; **Long.**: outputs at or above budget; **Avail.**: attempted-run structured-commitment availability. Covered metrics are on emitted commitments and are zero in every cell; **bold** marks the highest availability at each backend.

K Hy3-preview Output-Budget Diagnostic

Hy3-preview (Hunyuan3-preview) is not backend-comparable under the matched 2200-token budget: the formal run was stopped after 64/64 budget-exhausted JSON parse failures, and a low-concurrency check reproduced 3/3 invalid attempts. At 8800 tokens, Hy3-preview emits parseable commitments: one CBEA+LCV sample parses and validates, and a 12-fixture $\times$ 3-method diagnostic yields 27/27 valid runs before stopping. We treat this as an output-budget diagnostic only. It shows that reasoning-heavy endpoints can spend the matched budget before final JSON; the 8800-token run is small and not matched to the 360-fixture protocol.

L Shadow-Oracle Boundary Diagnostic

The shadow-oracle layer evaluates what falls outside validator scope. Each fixture includes hidden visible-fact labels, aliases, paraphrase patterns, weights, and turn-due markers; runtime prompts receive only user-visible observations and compiled fixture fields. Grep checks confirm that shadow fields never appear in runtime prompts. We score uncompiled-fact recall by deterministic alias and paraphrase matching over normalized output. Because CBEA selects only from compiled evidence E_t, this MiniMax-only diagnostic measures selector scope, not backend sensitivity.

Method	Avail.	Cov.H	Cov.E	Cov.W	Cov.C	Uncomp.
Raw	0.7889	0.1901	0.5246	0.1972	0.8415	0.5316
Raw + LCV	0.0361	0	0	0	0	0.3674*
Long-ctx + LCV	0.0500	0	0	0	0	0.2380*
Validator-only	0.4028	0.2000	0.4276	1.0000	1.0000	0.0000
CBEA + LCV	**0.4944**	0	0	0	0	**0.0119**

Table 18: Shadow-oracle boundary diagnostic over 360 MiniMax-M2.7 fixtures. Avail. uses attempted denominator; covered metrics are on emitted commitments. **Uncomp.**: uncompiled-fact recall. *Sparse denominator. CBEA+LCV preserves the compiled boundary at near-zero uncompiled recall; raw retains more uncompiled context while exhibiting high covered-boundary failures.

Domain	Raw	Raw+LCV	Long+LCV	Val-only	CBEA+LCV
investment	0.5733	–	0	0	0.0098
love_choice	0.5246	0.5193	0.2292	0	0.0053
career	0.4655	0.5000	0.2456	0	0.0132
relocation	0.5479	0	0	0	0.0133
comprehensive	0.5520	0.2222	0.5000	0	0.0187

Table 19: Per-domain uncompiled-fact recall (MiniMax-M2.7 shadow-oracle, 360 fixtures). CBEA+LCV is near zero across all five domains, confirming the selector blindness is structural and not domain-specific. Raw shows non-trivial cross-domain variance (0.47–0.57), confirming the matcher is not domain-saturated.

M Model-Judge Audit Details

Table 21 reports the open-weight model-judge audit in more detail. Judges received blinded system labels, the same rubric, and a plain-English runtime-control reference, but did not receive method names, automatic scores, raw production data, credentials, or oracle labels. This audit queried DeepSeek-V4-Pro and Qwen/Qwen3.6-35B-A3B through third-party inference endpoints. Both endpoints were used only to obtain blinded model-judge outputs. The model identifiers were `deepseek-v4-pro` and `qwen3.6-35b-a3b`. We report the identifiers, decoding settings, and judge prompts for auditability; exact outputs may vary with provider serving implementations. Judge calls used temperature 0, top-p 1, JSON response format, the fixed 0–2 rubric, and the same per-case winner-selection prompt for every held-out case.

Table 20 reports a case-cluster bootstrap over the 90 held-out cases, resampling cases with replacement and preserving both judge selections for each sampled case. Per-dimension means are descriptive: raw prompt stuffing receives higher surface-oriented scores on several dimensions, so Table 21 is not interpreted as a method-winner table.

Statistic	Observed	95% CI low	95% CI high
CBEA+LCV winner share	0.5000	0.4167	0.5833
Raw winner share	0.3056	0.2278	0.3833
Validator winner share	0.1778	0.1111	0.2500
Tie share	0.0167	0.0000	0.0389
CBEA–Raw margin	0.1944	0.0500	0.3389

Table 20: Case-cluster bootstrap over 180 model-judge winner selections from 90 cases and two judges. The CBEA–Raw margin is the difference in winner-selection share.

Judge	System	Overall	Evid.	Cons.	Surf.
DS	CBEA	1.6482	1.1333	1.3667	1.8333
DS	Raw	1.6704	1.2667	1.3889	1.8778
DS	Val.	1.4444	0.9111	0.8111	1.8333
QW	CBEA	1.6537	1.3111	1.3222	1.8000
QW	Raw	1.6852	1.3444	1.3444	1.8778
QW	Val.	1.3241	0.7222	0.5111	1.7444

Table 21: Per-judge descriptive means on a 0–2 rubric. DS is DeepSeek-V4-Pro, QW is Qwen/Qwen3.6-35B-A3B, CBEA is CBEA+LCV, Raw is raw prompt stuffing, and Val. is validator-only. These means are not the primary fidelity statistic and are not interpreted as method winners; they expose that raw prompt stuffing often receives higher surface/detail scores, while Table 20 reports the paired case-level winner diagnostic. Scores are not compared across judges.

Dimension	N	Exact	Within 1
Constraint fidelity	270	0.9222	0.9741
Evidence coverage	270	0.6593	0.9556
Consequence continuity	270	0.6630	0.9630
No-feasible handling	270	0.8111	0.9444
Appropriate personalization	270	0.9296	0.9741
Surface coherence	270	0.9185	0.9889
All dimension labels	1620	0.8173	0.9667
Case-level winner	90	0.6000	–

Table 22: Agreement between the two model judges. Exact agreement is strict equality on 0–2 labels or case-level winner.