FORKBENCH: Going Beyond Factuality to Evaluate Flow-based Conversational Hallucination in LLMs

Anonymous ACL submission

Abstract

Large language models (LLMs) deployed as conversational agents are increasingly expected to follow structured procedural logic. While factual hallucination has received extensive attention, a different failure mode has been overlooked: *conversational hallucination*, in which an agent produces individually fluent and factually plausible utterances while deviating from the intended conversational process. We introduce FORKBENCH, the first benchmark to study this failure mode systematically. FORKBENCH is constructed by *forking* well-formed dialogues at controlled turn positions and injecting one of four complementary hallucination types, each corrupting a different aspect of conversational structure, producing 36,278 dialogues across 24 tasks in 12 domains. We further define three evaluation tasks (set-to-set distance, dialog-to-dialog distance, and hallucination turn prediction) that characterize procedural deviation at three granularities, and propose two flow-aware distance metrics, FLOWKL and FLOWSCORE, that target *procedural* (dialog-flow) similarity, an axis distinct from semantic similarity. This opens new directions for procedural retrieval (retrieval by how a conversation unfolded, not its content), real-time conversational guardrails, and distributional monitoring of deployed agents. We release FORKBENCH under the MIT license, in the hope that the community will treat conversational *process* as measurable alongside conversational content.

1 Introduction

As LLMs are deployed in increasingly agentic roles (customer support, scheduling, guided troubleshooting, clinical intake), they are implicitly expected to follow structured procedural logic: a sequence of prescribed steps that moves a conversation from a user's initial request to a completed goal (Guo et al., 2024; Alismail and Lanquillon, 2025). This expectation extends well beyond classical task-oriented dialogues (TOD); it applies to any system where the *process* of the conversation matters as much as the content of individual utterances.

The dominant conception of hallucination in LLMs concerns *factual* errors (Maynez et al., 2020; Ji et al., 2023; Huang et al., 2025), for instance: confident assertions of incorrect dates, invented product details, or wrong numbers. A qualitatively different and practically critical failure mode has, however, received almost no attention: an agent that produces entirely truthful, fluent utterances while violating the procedural logic it was instructed to follow. The closest prior notions, hallucination in inter-agent communication (Lin et al., 2025) and faithfulness to grounding knowledge in dialogue (Dziri et al., 2022), concern the accuracy of *content*, not adherence to a conversational *process*.

We call this **conversational hallucination**: a breakdown in *process* rather than *content*. A conversationally hallucinating agent may skip required steps, pursue a different goal than instructed, repeat already-completed actions, or introduce conflicting objectives that derail task completion, all while generating individually plausible responses. Crucially, this failure is *orthogonal* to factual correctness: no single utterance need be false for the overall conversation to be wrong. Standard hallucination benchmarks (Li et al., 2023; Lin et al., 2022) are blind to such failures because they evaluate the truthfulness of content, not the correctness of the conversational process.

To address this gap, we present FORKBENCH, a benchmark constructed by *forking* dialogues at controlled turn positions (Figure 1) to simulate different types of conversational hallucinations. Our main contributions are the following: (a) We define *conversational hallucination* as a phenomenon distinct from factual hallucination, and decompose it into four complementary types grounded in a general workflow model (§3). (b) We release FORK-

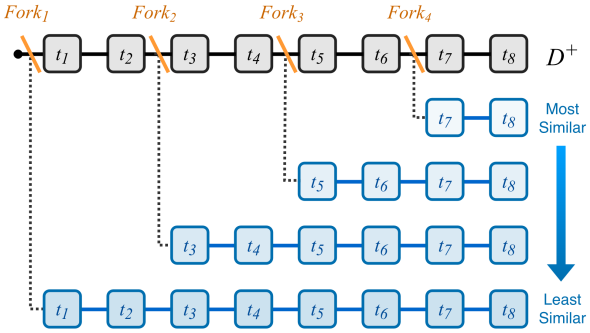

Figure 1: Construction of FORKBENCH by controlled forking. To produce a forked variant of a conformant dialogue D^+, the simulation is re-run with the same random seed (yielding an identical prefix), and a hallucination instruction is injected into the system agent's context just before the fork turn, causing the continuation to diverge. Earlier forks share less prefix and yield more deviant dialogues. This induces a natural ground-truth ordering by *procedural* (dialog-flow) similarity, distinct from pure semantic similarity.

BENCH, a 36,278-dialogue benchmark constructed via controlled forking that spans 24 tasks across 12 domains and four temporal deviation depths, along with the seed-controlled multi-agent generation methodology used to produce it, which can be applied to any workflow-annotated dialogue dataset (§4). (c) We define three evaluation tasks (set-to-set distance, dialog-to-dialog distance, and hallucination turn prediction) that, together, characterize procedural deviation at three granularities not measurable before (§5). (d) Enabled by the new dataset, we introduce FLOWKL and FLOWSCORE, two flow-aware distance metrics that exploit the procedural structure FORKBENCH exposes, and report an initial set of baseline results across all three tasks as a starting point for future work (§6).

2 Related Work

Hallucination in LLMs. The dominant view concerns factual accuracy of generated content (Maynez et al., 2020; Kryściński et al., 2020; Lin et al., 2022; Mallen et al., 2023). Surveys (Ji et al., 2023; Zhang et al., 2025b; Huang et al., 2025) taxonomize hallucination along faithfulness and factuality dimensions; all existing categories address the truthfulness of *what* is said, not the correctness of *how* a conversation unfolds.

Procedural adherence and dialogue similarity. A parallel line studies whether systems follow prescribed steps: compliance with complex condition–action instructions in TOD (Ghazarian et al., 2025) or the generation of structured, possibly non-linear workflows (Qiao et al., 2025). These assess whether a system *selects* correct actions, typically over symbolic TOD components such as intents, slots, or dialogue acts (Bhaumik et al., 2023). Dialogue similarity, in turn, has largely been treated as text similarity applied to dialogues: sentence or dialogue embeddings collapse the sequence of conversational turns into a single vector (Reimers and Gurevych, 2019; Zhou et al., 2022), followed by cosine similarity, thereby discarding interaction dynamics and turn ordering. None of these capture *how* a conversation unfolds. FORKBENCH differs in two ways: it characterizes how a conversation *deviates* from a conformant process rather than scoring action selection, and it *operates directly on raw dialogue text*, with no TOD-specific components, and applies to any goal-directed conversation.

3 Conversational Hallucination

3.1 Formal Model

Most conversational systems do not declare their procedural logic explicitly: states, available actions, and transitions are implicit in the agent's instructions, the underlying business process, or the assumed conversational norms. To talk about *deviations* from this logic precisely, we introduce a lightweight abstraction that does not assume any particular implementation. We model the intended logic of a conversational agent as a tuple $\mathcal{W} = (\mathcal{S}, \mathcal{A}, \delta, \mathcal{O})$, where $\mathcal{S}$ is a set of conversational states (*e.g.,* steps or phases of the interaction), $\mathcal{A}$ is a set of agent actions (*e.g.,* the communicative functions it can perform), $\delta : \mathcal{S} \times \mathcal{U} \to \mathcal{S}$ is a transition function from the current state and user input $\mathcal{U}$ to the next state, and $\mathcal{O}$ is the agent's objective.

This formulation is intentionally general. It applies to any goal-directed conversational agent. For example, a task-oriented dialogue system has the dialogue policy as δ and task completion (*e.g.,* booking a hotel room, retrieving a balance) as $\mathcal{O}$; a medical-intake assistant has a clinical protocol as δ and patient triage as $\mathcal{O}$; a customer-support agent has a troubleshooting procedure as δ and issue resolution as $\mathcal{O}$. We define a *conformant* dialogue D^+ as one whose turns are produced in accordance with $\mathcal{W}$.

3.2 Hallucination Types

We define four complementary types, each targeting a distinct component of $\mathcal{W}$ (Table 1):

Table 1: Conversational hallucination types and the components of $\mathcal{W} = (\mathcal{S}, \mathcal{A}, \delta, \mathcal{O})$ they corrupt. ✓: preserved; ×: replaced; ∼: partially overridden.

Type	$(\mathcal{S}, \mathcal{A})$	δ	$\mathcal{O}$	What is broken
WIC	×	×	✓	Workflow identity
SSPC	✓	×	✓	Step selection
GR	✓	✓	×	Task goal
CSC	✓	✓	∼	Goal conflict

WIC— Workflow Identity Corruption. $\mathcal{W}$ is replaced by a different workflow $\mathcal{W}'$; the agent follows a valid workflow but the wrong one (modeling task disambiguation failures and prompt injection).

SSPC— Step Selection Policy Corruption. $(\mathcal{S}, \mathcal{A})$ and $\mathcal{O}$ are preserved but δ is replaced by a corrupted policy δ' that, *e.g.,* repeats, skips, loops, or randomizes steps.

GR— Goal Replacement. $\mathcal{W}$ is preserved but the objective $\mathcal{O}$ is replaced by a qualitatively different goal $\mathcal{O}'$ (*e.g.,* collecting feedback instead of completing a booking).

CSC— Conflicting System Constraints. $\mathcal{O}$ is retained but augmented with a competing auxiliary objective $\mathcal{O}_c$ that intermittently overrides $\mathcal{O}$ (*e.g.,* always responding with a question, or confirming every detail).

In practice, for both diversity and fine-grained controllability, we define a small pool of concrete variants per hallucination type (five corrupted policies for SSPC such as *repeat*, *skip*, and *invert order*, six alternative goals for GR, and six conflicting constraints for CSC); the full lists are in Appendix B.

4 The FORKBENCH Dataset

4.1 Construction Pipeline

STAR as a generation substrate. We base FORKBENCH on STAR (Mosig et al., 2020), which spans 24 tasks across 12 domains (hotel, ride, restaurant, medical, party, spaceship, trivia, weather, etc.). STAR is uniquely suited to our pipeline because its dialogues were originally produced by *human role-play*: two annotators were given a system persona together with a workflow graph and per-node response templates,[1] and a user persona with a task and goal, and were asked to converse as the system and the user respectively. Crucially, STAR releases not only the resulting dialogues but also the per-dialogue scenario metadata that was given to each pair of annotators (see Appendix F for a complete example), which we re-use to drive LLM role-play. In this instantiation, $\mathcal{S}$ corresponds to STAR's workflow nodes, $\mathcal{A}$ to system responses, δ to the directed workflow graph, and $\mathcal{O}$ to the task-completion goal specified in each scenario.

Conformant dialogue generation. For each STAR scenario, we use SDIALOG (Burdisso et al., 2026) to instantiate a two-agent simulation in which two LLMs receive the *exact same role-play metadata* that was given to the human annotators: a **system agent** is given the system persona, workflow graph, and response templates and is constrained to follow the workflow faithfully; a **user agent** is given the user persona, task, and goal and is instructed to interact naturally without volunteering all details upfront. Both agents are `Gemma4-31B` (Google DeepMind, 2026) served locally[2]; prompts and full implementation details are in Appendix A. The two agents converse until the task is completed or a per-task turn budget is reached. After filtering for dialogues that include enough system turns to admit all four fork positions, we retain an initial set of **2,134 valid conformant dialogues**.

Forked hallucination generation via prefix-preserving replay. For each conformant dialogue D^+ with T system turns, we identify four *fork points* at evenly-spaced positions $t_k = 1 + \lfloor (k-1) \cdot T/4 \rfloor$, $k \in \{1, 2, 3, 4\}$. To produce a forked variant, we re-run the same scenario under fixed-seed determinism so that turns 1 through $t_k - 1$ are identical to those of D^+; before turn t_k is generated, a hallucination instruction is added as a system message to the system agent's context, specifying the type and the sampled variant (drawn from the per-type pool defined in §3; full prompts in Appendix C). From t_k onward, the system agent follows the injected instruction and produces turns that diverge from D^+ in the manner specified by the chosen hallucination type (see Appendix H for worked examples). Earlier forks produce more deeply deviant dialogues, while later forks produce subtler deviations , naturally inducing a similarity ranking over the forked dialogues (Figure 1).

[1] See Appendix E for a complete example of STAR's workflow metadata.

[2] Selected after manually inspecting pilot dialogues from several open-weight LLMs (`Gemma3` and `4`, `Qwen2.5`, `3`, and `3.6`, `Phi4`, `DeepSeek-R1`, `Nemotron-3-Nano`, `Llama3.1`) and from `GPT-4.1` as a closed-source reference.

Table 2: Dataset composition across conformant and forked splits, with per-split coverage and average dialogue length in turns (#Turns) and words (#Words).

	Coverage		Dialog Length	
Split	**#Dialogs**	**%Total**	**#Turns**	**#Words**
Conformant	2,134	5.88%	15.61	203.84
WIC forked	8,536	23.53%	20.01	293.61
SSPC forked	8,536	23.53%	19.25	229.74
GR forked	8,536	23.53%	24.19	377.52
CSC forked	8,536	23.53%	20.54	268.02
Total	**36,278**	**100.00%**	**20.68**	**287.02**

Quality assurance. FORKBENCH's usefulness rests on a single property: *forking must produce controlled deviation from the conformant baseline*. We verify this property in two stages. First, four authors of this paper jointly inspected a stratified sample of over 400 dialogues across all 24 tasks, confirming that conformant dialogues followed the intended workflow and that forked dialogues exhibited the intended hallucination from the fork turn onward. More importantly, we designed a set-to-set ranking evaluation (§5) precisely to verify this property at the scale of the entire dataset by measuring the distance between the set of conformant dialogues and each set of forked dialogues at increasing fork positions: as we will see in §6, every metric we evaluated correctly recovers the expected ordering across all the tasks and hallucination types, certifying that forking does indeed produce the controlled deviation FORKBENCH's design rests on.

4.2 Dataset Statistics

Each of the 2,134 conformant dialogues yields 16 forked variants (4 hallucination types $\times$ 4 fork positions), for a total of **36,278 dialogues** (Table 2). The table also reveals that forked dialogues are systematically longer than their conformant counterparts (21.00 vs. 15.61 turns on average), with the effect varying by hallucination type: GR dialogues are the longest (24.19 turns on average), reflecting the verbose conversational behavior induced by replacing the task objective with goals such as information collection or user education; SSPC dialogues are the shortest forked variants (19.25 turns), consistent with workflows that skip or repeat steps rather than introducing new content. Table 3 provides a complementary per-task coverage snapshot. Coverage varies substantially across tasks, reflecting both the heterogeneous yield of valid conformant generations and the variable workflow complexity inherited from STAR; per-task dialogue lengths span more than 2$\times$ (from 12.25 turns for *ride_status* to 29.71 for *trivia*). A full split-aware per-task breakdown is reported in Appendix G.

Table 3: Per-task coverage and average dialogue length in turns (#Turns) and words (#Words) in FORKBENCH. Detailed split-aware statistics are in Appendix G.

	Coverage		Dialog Length	
Task	**#Dial.**	**%Tot.**	**#Turns**	**#Words**
apartment_schedule	3,502	9.65%	21.94	317.82
apartment_search	510	1.41%	16.06	328.47
bank_balance	782	2.16%	17.84	220.89
bank_fraud_report	2,295	6.33%	19.58	253.54
doctor_followup	714	1.97%	13.22	178.18
doctor_schedule	2,125	5.86%	21.57	292.31
hotel_book	2,074	5.72%	17.62	205.62
hotel_search	1,411	3.89%	13.43	210.64
hotel_service_req.	782	2.16%	20.18	256.21
meeting_schedule	697	1.92%	19.25	242.15
trip_directions	3,536	9.75%	27.03	389.40
party_plan	3,145	8.67%	24.86	340.64
party_rsvp	2,159	5.95%	22.08	298.15
plane_book	1,530	4.22%	17.46	188.42
plane_search	493	1.36%	18.16	239.16
restaurant_book	459	1.27%	20.71	263.83
restaurant_search	748	2.06%	15.72	263.15
ride_book	2,329	6.42%	17.68	230.42
ride_change	221	0.61%	12.93	149.27
ride_status	153	0.42%	12.25	145.32
spaceship_access	1,122	3.09%	19.66	306.96
spaceship_life	680	1.87%	13.77	247.97
trivia	2,788	7.69%	29.71	462.15
weather	2,023	5.58%	13.80	161.59
TOTAL	**36,278**	**100%**	**20.68**	**287.02**

5 Evaluation Tasks and Methods

We define three evaluation tasks on FORKBENCH at increasing levels of granularity: set-level, dialogue-level, and turn-level. The first two probe procedural (dialog-flow) deviation as a measurable distance, an axis distinct from semantic similarity: two dialogues can be near-identical semantically (*e.g.,* same topics, same vocabulary) yet follow entirely different conversational procedures, and vice versa. The third localizes the deviation, asking where in a dialogue a hallucination first manifests.

5.1 Task 1: Set-to-Set Distance Ranking

For each hallucination type h and each of the 24 tasks of FORKBENCH, we group the forked dialogues by their fork position to obtain four sets $\mathcal{D}_k^h$ ($k \in \{1, 2, 3, 4\}$), and take the set of conformant dialogues for that task, $\mathcal{D}^+$, as the reference. Writ-

ing $\mu_k \equiv \mu(\mathcal{D}^+, \mathcal{D}_k^h)$ for the distance under set-distance metric μ, we expect the natural *procedural-similarity* ordering $\mu_1 > \mu_2 > \mu_3 > \mu_4$, since earlier forks share less of the conformant prefix and are therefore more procedurally deviant. We report Kendall's τ (Kendall, 1938) between the metric-induced ranking and this ground-truth ordering, averaged across the 24 tasks. This task primarily serves as a *sanity check* that the dataset construction is sound.

Baselines. To the best of our knowledge, FBD and PRD, introduced together by Xiang et al. (2021), are the only two existing distribution-wise metrics specifically designed to compare sets of dialogues. Both operate on embeddings of consecutive (system, user) turn pairs extracted by a pretrained language model. **FBD** (Fréchet BERT Distance) fits a multivariate Gaussian to each set's embedding distribution and reports the Fréchet distance between the two Gaussians. **PRD** (Precision–Recall Distance) treats the two sets as distributions and reports the maximum harmonic mean of precision and recall over a sweep of relative-density thresholds (Sajjadi et al., 2018).

FLOWKL. We propose FLOWKL, an interpretable alternative grounded in an explicit dialog flow graph G.

1. **Graph construction.** Following Burdisso et al. (2024), we cluster all utterances in $\mathcal{D}^+$ in the Dialog2Flow (D2F) embedding space[3] to obtain workflow-step nodes; directed edges encode observed transitions with frequency weights, yielding a bigram model $P_G(n_t \mid n_{t-1})$ over node sequences (Figure 3 in Appendix shows the actual graph induced for the *doctor_schedule* task).
2. **Per-dialogue perplexity with node confidence.** For a dialogue $d = (u_1, \ldots, u_T)$, we encode each turn as $\phi(u_t)$ in D2F space and assign its node n_t as the nearest centroid. Because the assignment may be uncertain when $\phi(u_t)$ lies between clusters, we weight each transition by defining a *node confidence* (NC),

$$P_{\text{NC}}(n_t \mid u_t) = \frac{\exp(s_1)}{\sum_{k=1}^{K} \exp(s_k)},$$

   where s_k is the cosine similarity between $\phi(u_t)$ and the k-th nearest centroid. Since D2F is trained with a contrastive InfoNCE objective (Oord et al., 2018), this softmax approximates the posterior that u_t belongs to n_t (i.e., that n_t's centroid is the *positive* for u_t) given the $K-1$ neighbors as *negatives*. The NC-weighted transition is then $P(n_t \mid n_{t-1}) = P_G(n_t \mid n_{t-1}) \cdot P_{\text{NC}}(n_t \mid u_t)$, and the per-dialogue flow perplexity is the geometric mean of $-\log P(n_t \mid n_{t-1})$ over the dialogue.
3. **Distribution distance.** We kernel-density-estimate the per-dialogue perplexities of each set and compute the KL divergence $\text{KL}(p_{\text{ref}} \,\|\, p_{\text{tgt}})$.

We also evaluate an ablation without the NC factor. FLOWKL yields a single interpretable scalar (perplexity over G) that summarizes how well a set of dialogues conforms to the induced reference workflow G, which is itself human-readable.

5.2 Task 2: Dialog-to-Dialog Distance Ranking

Task 2 probes procedural (dialog-flow) similarity at the level of individual dialogues. For each conformant dialogue D^+, the four forked dialogues D_k^h provide an unambiguous 4-item ranking of distance from D^+ (Figure 1), and a good dialogue-distance metric should recover this ranking. We report Kendall's τ between predicted and ground-truth rankings, averaged across all conformant dialogues. Unlike Task 1, to the best of our knowledge, no metric has been specifically designed to measure dialogue-to-dialogue distance; we therefore compare standard pooling strategies over per-turn embeddings against two new flow-aware alternatives. Both flow-aware methods operate on the per-turn embedding sequences $E = (e_1, \ldots, e_n)$ and $E' = (e'_1, \ldots, e'_m)$ of D^+ and D_k^h respectively.

Pooling baselines. **Mean-turn** averages per-turn embeddings; **Full-dialog** encodes the concatenated transcript once. Both use cosine distance.

FLOWDTW. We align E and E' using Dynamic Time Warping (DTW) (Sakoe and Chiba, 1978). Local costs are $c(i,j) = 1 - e_i^\top e'_j$ and the DTW recurrence $D(i,j) = c(i,j) + \min\{D(i-1,j), D(i,j-1), D(i-1,j-1)\}$ with $D(0,0) = 0$ gives the optimal aligned cost; the final distance is path-length normalized: $d_{\text{FlowDTW}} = D(n,m)/L$ for path length L.

FLOWSCORE. A BERTScore-inspired (Zhang et al., 2020) matching applied at the turn level. We

[3] D2F (Burdisso et al., 2024) is a sentence-embedding model trained to cluster utterances by their communicative function rather than surface form.

build $S(i,j) = e_i^\top e'_j$ and compute

$$P = \frac{1}{n}\sum_i \max_j S(i,j), \quad R = \frac{1}{m}\sum_j \max_i S(i,j),$$

$F_{\text{flow}} = 2PR/(P+R)$, and report $d_{\text{FlowScore}} = 1 - F_{\text{flow}}$. FlowScore captures soft turn-level alignment without requiring strict monotonic warping of DTW.

5.3 Task 3: Hallucination Turn Prediction

Task 3 probes whether a method can localize *where* a dialogue first goes off-track. For conformant dialogues, the system should not flag any turn; for forked dialogues, the predictor's goal is to identify the fork point t_k as closely as possible. We consider three regimes of decreasing realism:

1. **Streaming.** Turn-by-turn inference, calibrated on conformant dialogues only. The most realistic and hardest setting.
2. **Non-streaming.** Inference after observing the full dialogue, without workflow access.
3. **Non-streaming + workflow.** Inference with the full dialogue *and* the ground-truth workflow. Typically unavailable in practice; reported as an upper-bound reference.

We evaluate the following methods in each regime:

Streaming. *First*, for each dialogue we map turn embeddings to discrete nodes via the encoder's clustering,[4] and fit an N-gram language model over the conformant node sequences. Let $p_t = P_{\text{LM}}(n_t \mid n_{t-N+1}, \ldots, n_{t-1})$ and $p_t^{\text{NC}} = p_t \cdot c_t$, where $c_t \in [0,1]$ is the top-1 node-assignment confidence (as in FlowKL). *Then*, on top of this LM, we define two detectors. **Improbable Transition (IT)**: predict the first t with $p_t < \theta_{\text{off}}$ (or $p_t^{\text{NC}} < \theta_{\text{off}}$ in the NC variant); θ_{off} is the 5th percentile of $\min_t p_t$ over conformant dialogues (target false-alarm rate $\approx 5\%$). **Surprisal Spike (SS)**: let $s_t = -\log p_t$ and $\Delta_t = s_t - s_{t-1}$; predict the first t with $\Delta_t > \theta_{\text{jump}}$, calibrated as the 95th percentile of $\max_t \Delta_t$ over conformant dialogues.

Non-streaming. In the non-streaming regimes we use zero-shot LLM prompting: the LLM is asked to identify the system turn where the dialogue first deviates from the intended workflow (or to indicate that no deviation occurs). Without workflow access, the prompt includes only the dialogue; with workflow access, the ground-truth workflow is also provided. The exact prompts are in Appendix D.

[4]The same recipe used to build the nodes in G for FlowKL.

Table 4: Task 1 (Set-to-Set Distance): Kendall's τ between predicted and ground-truth fork-position rankings, averaged across the 24 tasks. Higher is better; **bold** marks the best per column. *w/o* NC is the ablation that removes node-confidence weighting from FlowKL.

Method	WIC	SSPC	GR	CSC	Avg.
FBD	**0.993**	0.910	0.979	**0.951**	**0.958**
PRD	**0.993**	0.889	**0.986**	0.861	0.932
FlowKL	0.979	**0.958**	0.951	0.931	0.955
w/o NC	0.965	0.924	0.972	0.931	0.948

Beyond the per-regime methods, we also include three trivial baselines and report the metrics below.

Baselines. Three predictors that always predict *first turn*, *last turn*, or a *random* turn.

Metrics. For hallucinated dialogues, we report the mean absolute error (MAE) of the predicted turn against the fork turn and the *hallucinated coverage*: the fraction of forked dialogues flagged. For conformant dialogues, we report the MAE relative to the dialogue end and the *conformant coverage* (the false-alarm rate, to be minimized). A composite Score combines these into a single ranking value (Appendix I).

6 Experiments

6.1 Task 1: Set-to-Set Distance Ranking

Table 4 reports set-to-set ranking results. All four methods exceed $\tau = 0.90$ on average, confirming that the fork-based construction yields a well-formed deviation ordering across the entire dataset.

Two findings deserve emphasis. First, FlowKL matches FBD within 0.003 on average (0.955 vs. 0.958) while operating on a single interpretable scalar (perplexity) over a human-readable graph (*e.g.,* Figure 3) rather than on high-dimensional embedding distributions. Second, on SSPC (the hallucination type that explicitly targets step ordering), FlowKL reaches τ=0.958 vs. FBD's 0.910, where an explicit transition model captures step ordering better than FBD/PRD's implicit distributions over consecutive-turn embeddings. The NC ablation isolates a small but consistent benefit on flow-sensitive types (WIC, SSPC) with a marginal trade-off on GR, suggesting NC helps when am-

Table 5: Task 2 (Dialog-to-Dialog Distance): Kendall's τ between predicted and ground-truth fork rankings, averaged across all conformant dialogues. Higher is better; **bold** marks the best method within each embedding model and **bold+underline** the best overall. Mean = mean-turn embedding; Full = full-dialog embedding.

Encoder	Method	WIC	SSPC	GR	CSC	Avg.
LaBSE	Mean	0.768	0.786	0.854	0.596	0.751
	Full	0.851	0.778	0.857	0.604	0.772
	FlowDTW	0.857	0.794	0.896	0.583	0.782
	FlowScore	**0.939**	**0.883**	**0.928**	**0.774**	**0.881**
Qwen3-Emb.	Mean	0.770	0.670	0.780	0.455	0.669
	Full	0.589	0.598	0.663	0.317	0.542
	FlowDTW	0.858	0.776	0.872	0.531	0.759
	FlowScore	**0.917**	**0.860**	**0.895**	**0.702**	**0.844**
SBERT	Mean	0.767	0.748	0.813	0.558	0.722
	Full	0.684	0.714	0.731	0.525	0.663
	FlowDTW	0.854	0.787	0.882	0.605	0.782
	FlowScore	**0.941**	**0.898**	**0.917**	**0.769**	**0.881**
BGE-M3	Mean	0.778	0.760	0.839	0.555	0.733
	Full	0.780	0.780	0.832	0.584	0.744
	FlowDTW	0.840	0.789	0.879	0.569	0.769
	FlowScore	**0.936**	**0.889**	**0.925**	**0.756**	**0.876**
D2F	Mean	0.809	0.770	0.864	0.574	0.754
	Full	0.809	0.787	0.809	0.595	0.750
	FlowDTW	0.873	0.762	0.901	0.591	0.782
	FlowScore	**0.927**	**0.874**	**0.926**	**0.738**	**0.866**
DSE	Mean	0.780	0.752	0.844	0.553	0.732
	Full	0.789	0.767	0.789	0.575	0.730
	FlowDTW	0.859	0.753	0.883	0.594	0.772
	FlowScore	**0.939**	**0.883**	**0.919**	**0.757**	**0.875**
Emb.Gemma	Mean	0.762	0.741	0.837	0.552	0.723
	Full	0.749	0.758	0.831	0.542	0.720
	FlowDTW	0.848	0.765	0.881	0.567	0.765
	FlowScore	**0.930**	**0.882**	**0.919**	**0.765**	**0.874**

biguous node assignments would otherwise propagate misleading transitions.

6.2 Task 2: Dialog-to-Dialog Distance Ranking

Table 5 reports the dialog-to-dialog ranking results across seven encoder families, covering general-purpose sentence embeddings (SBERT (Reimers and Gurevych, 2019), LaBSE (Feng et al., 2022), BGE-M3 (Chen et al., 2024), Qwen3-Embedding (Zhang et al., 2025a), EmbeddingGemma (Vera et al., 2025)) and dialogue-specific embeddings (DSE (Zhou et al., 2022) and D2F (Burdisso et al., 2024)). For each family we report the best-performing variant as representative: `all-mpnet-base-v2`[5] for SBERT, `Qwen3-Embedding-8B`[6] for Qwen3, `dse-bert-base`[7] for DSE, and `dialog2flow-joint-dse-base`[8] for D2F. Per-task breakdowns appear in Table 9 (Appendix).

FlowScore is consistently the strongest method across all seven encoder families, lifting every encoder to average $\tau > 0.85$. This robustness suggests the method's strength is structural rather than encoder-specific: even when the underlying encoder is weaker at capturing procedural similarity, the turn-level matching recovers most of the signal. The clearest illustration is Qwen3-Embedding, a strong semantic encoder whose Full-dialogue pooling collapses to $\tau = 0.542$ (the lowest baseline result in the table), yet FlowScore still reaches $\tau = 0.844$ on the same encoder. This illustrates a key point: *semantic* similarity and *procedural* similarity are distinct axes, and even strong semantic encoders can be poor procedural-similarity proxies on their own.

6.3 Task 3: Hallucination Turn Prediction

Table 6 reports results for all methods and regimes. For non-streaming zero-shot prompting (§5.3), we use three recent open-weight models from major labs at comparable scale: `Gemma4-31B` (Google DeepMind, 2026), `Qwen3.6-35B` (Qwen Team, 2026), and `GPT-OSS-20B` (OpenAI et al., 2025). For streaming, we report only the best-performing configuration of the N-gram language model over turn nodes (*i.e.,* $N{=}6$ with the D2F encoder).[9] The full per-encoder breakdown is available in Appendix J. **Streaming results.** Despite operating turn-by-turn with no access to future context and using only conformant dialogues for calibration, our streaming methods achieve hallucinated-MAE around 5 turns while keeping the false-alarm rate (conformant coverage) at only ∼4–6%. IT marginally outperforms SS in raw MAE; NC weighting consistently improves performance for both, with the best variant (SS+NC) reaching Score 72.32. **Non-streaming.** The three LLMs see substantial gains over streaming on raw MAE, but hallucinated coverage drops sharply: `Gemma4-31B` flags only 71.35% of forked dialogues vs. streaming's 93.93%, indicating substantially more abstentions on hallucinated inputs. Its composite Score (83.63) clearly exceeds the best streaming method (72.32), with the gap driven by sharper localization (MAE 2.82 vs. 5.07) and lower conformant

[5] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[6] https://huggingface.co/Qwen/Qwen3-Embedding-8B

[7] https://huggingface.co/aws-ai/dse-bert-base

[8] https://huggingface.co/sergioburdisso/dialog2flow-joint-dse-base

[9] $N{=}6$ was selected after a sweep over $\{2, 3, 6, 10\}$, where Score peaked at $N{=}6$.

Table 6: Task 3 (Turn Prediction): main results. MAE and conformant coverage (Cov.): lower is better. Hallucinated coverage and Score: higher is better. **Bold**: best within a regime; **bold+underline**: best overall. Baselines are not highlighted.

	Hallucination MAE ↓				Hallucination		Conformance		
Method	**CSC**	**GR**	**SSPC**	**WIC**	**Avg. MAE ↓**	**Cov. ↑**	**MAE ↓**	**Cov. ↓**	**Score ↑**
Baselines									
Random	6.63	8.24	6.20	6.93	7.00	100.00	8.05	100.00	0.94
First turn	2.83	2.83	2.83	2.83	2.83	100.00	14.02	100.00	0.00
Last turn	14.56	17.61	13.23	14.80	15.05	100.00	1.00	100.00	0.00
Streaming									
Surprisal Spike (SS)	7.13	4.63	4.27	5.03	5.27	93.05	5.27	**4.17**	71.70
w/ NC	7.07	4.61	4.24	4.98	5.22	93.48	**5.04**	5.62	**72.32**
Improbable Transition (IT)	7.05	4.27	4.23	4.93	5.12	93.33	5.36	**4.17**	71.81
w/ NC	**6.95**	**4.23**	**4.21**	**4.87**	**5.07**	**93.93**	5.29	5.48	72.08
Non-streaming									
`GPT-OSS-20B`	5.00	3.42	2.87	2.22	3.38	67.96	2.32	15.28	79.04
`Qwen3.6-35B`	**4.20**	3.38	2.23	2.36	3.04	**83.34**	2.90	29.52	78.67
`Gemma4-31B`	5.04	**3.06**	**1.23**	**1.96**	**2.82**	71.35	**1.66**	**7.31**	**83.63**
Non-streaming + workflow									
`GPT-OSS-20B`	2.93	1.83	1.39	1.47	1.91	**96.02**	2.76	82.05	67.85
`Qwen3.6-35B`	2.80	1.89	1.62	1.27	1.89	95.31	2.80	51.17	78.68
`Gemma4-31B`	**2.58**	**0.34**	**0.65**	**0.69**	**1.06**	92.37	**2.59**	**19.54**	**86.80**

MAE. **Non-streaming + workflow.** When LLMs additionally receive the ground-truth workflow, performance improves further: `Gemma4-31B` reaches Score 86.80 and an average hallucinated-MAE of only 1.06 turns, near-perfect localization. The gain comes with markedly higher conformant coverage (19.54% false-alarm rate for `Gemma4-31B`; up to 82.05% for `GPT-OSS-20B`), suggesting that in our setting, providing the workflow makes the tested models more likely to flag conformant dialogues as well.

While streaming methods do not match the raw MAE of large LLMs, they remain attractive in practice: orders of magnitude cheaper, real-time, and requiring no workflow access. All three regimes still have substantial headroom for future work, with streaming the most open and the most operationally relevant.

7 Conclusions

LLMs deployed in agentic conversational roles can produce individually plausible turns while violating the procedural logic the conversation was meant to follow. We named this failure mode *conversational hallucination* and introduced FORKBENCH, the first benchmark that makes it systematically measurable. More broadly, this surfaces an axis of conversational quality that existing tools largely miss: *procedural* similarity, distinct from the *semantic* similarity that dominates current evaluation. Even strong semantic encoders are poor procedural-similarity proxies on their own.

We hope FORKBENCH motivates more than a single benchmark. Each of the three tasks corresponds to a broader class of downstream applications that remains underexplored from a procedural perspective: set-to-set distance (Task 1) enables distributional monitoring of deployed agents and principled A/B testing between variants; dialog-to-dialog distance (Task 2) enables *procedural retrieval*, retrieving past conversations by how they unfolded rather than what they were about, a missing primitive for case-based reasoning and in-context example selection in agentic systems; hallucination turn prediction (Task 3) enables real-time guardrails that intervene at the turn an agent goes off-track, and active learning loops that prioritize procedurally-deviant examples for annotation.

The released generation pipeline makes no STAR-specific assumptions and can construct FORKBENCH-style benchmarks for any domain, language, or task family. We release the dataset and the code under the MIT license and hope the community will use FORKBENCH to push past these initial baselines, treating conversational *process* as measurable alongside conversational content.

Limitations

Single-source substrate. FORKBENCH is built on STAR, which spans 24 tasks across 12 domains. While the construction pipeline and the formal model make no STAR-specific assumptions and apply to any dialogue corpus with workflow-level metadata, the empirical results we report are necessarily tied to STAR's distribution of tasks, domains, and conversational styles. Extending FORKBENCH to other substrates (non-English dialogues, dialogues from non-TOD domains such as medical intake or customer support, etc.) is left to future work and to users of the released generation pipeline.

Single-type hallucinations. Each forked dialogue injects exactly one hallucination type at one fork point. Real conversational failures can be compound (multiple types co-occurring) or cascading (one corruption inducing another over subsequent turns). The detectors we propose are not evaluated under such combined conditions, and the four-type taxonomy may need extension to cover them.

Simulated rather than observed dialogues. Both conformant and forked dialogues are generated by LLM role-play (`Gemma4-31B`) under controlled prompting. This gives us scale, reproducibility, and exact ground-truth fork points, but the resulting distribution of language and conversational behavior may differ from dialogues collected from real users and deployed agents. Methods that work well on FORKBENCH may behave differently on naturally-occurring conversations, and conversely.

Baseline scope. The methods we report are intended as initial baselines, not as the strongest possible detectors. The streaming detectors use simple N-gram models over discrete nodes with no neural training beyond the pretrained encoder; the non-streaming detectors use zero-shot prompting of three open-weight LLMs without fine-tuning, few-shot examples, or chain-of-thought elicitation. Substantially stronger detectors are likely possible on all three tasks, and we hope users of FORKBENCH will explore them.

8 Ethical Considerations

FORKBENCH is constructed entirely from synthetically generated dialogues; no real user data, personal information, or human-subject interaction is involved, so the benchmark raises no privacy or consent concerns. Two points nonetheless warrant note. Since the dialogues are produced by an LLM, they may reflect biases present in the generator, and the 24 service-oriented tasks are not necessarily a representative sample of real-world conversational settings; results on FORKBENCH should be read as evidence about procedural deviation under controlled conditions rather than as direct estimates of deployed-system behavior. We judge the value of an openly available, reproducible benchmark to outweigh this risk, particularly because the same machinery enables the detection, guardrail, and monitoring directions the work is intended to support. FORKBENCH and all accompanying code are released under the MIT license; their intended use is research on conversational reliability, consistent with that license.

References

Ahmad Alismail and Carsten Lanquillon. 2025. A survey of llm-based methods for synthetic data generation and the rise of agentic workflows. In *Artificial Intelligence in HCI*, pages 119–135, Cham. Springer Nature Switzerland.

Ankita Bhaumik, Praveen Venkateswaran, Yara Rizk, and Vatche Isahagian. 2023. TaskDiff: A similarity metric for task-oriented conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16234–16240, Singapore. Association for Computational Linguistics.

Sergio Burdisso, Séverin Baroudi, Yanis Labrak, David Grünert, Pawel Cyrta, Yiyang Chen, Srikanth Madikeri, Esaú Villatoro-Tello, Ricard Marxer, and Petr Motlicek. 2026. SDialog: A python toolkit for end-to-end agent building, user simulation, dialog generation, and evaluation. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 320–340, Rabat, Marocco. Association for Computational Linguistics.

Sergio Burdisso, Srikanth Madikeri, and Petr Motlicek. 2024. Dialog2Flow: Pre-training soft-contrastive action-driven sentence embeddings for automatic dialog flow extraction. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5421–5440, Miami, Florida, USA. Association for Computational Linguistics.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*,

pages 2318–2335, Bangkok, Thailand. Association for Computational Linguistics.

Nouha Dziri, Ehsan Kamalloo, Sivan Milton, Osmar Zaiane, Mo Yu, Edoardo M. Ponti, and Siva Reddy. 2022. FaithDial: A faithful benchmark for information-seeking dialogue. *Transactions of the Association for Computational Linguistics*, 10:1473–1490.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891. Association for Computational Linguistics.

Sarik Ghazarian, Abhinav Gullapalli, Swair Shah, Anurag Beniwal, Nanyun Peng, Narayanan Sadagopan, and Zhou Yu. 2025. TOD-ProcBench: Benchmarking complex instruction-following in task-oriented dialogues. *Preprint*, arXiv:2511.15976.

Google DeepMind. 2026. Gemma 4 31b. https://huggingface.co/google/gemma-4-31B. Accessed 2026-05-24.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 8048–8057. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Trans. Inf. Syst.*, 43(2).

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Maurice G. Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464, Singapore. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Xixun Lin, Yucheng Ning, Jingwen Zhang, Yan Dong, Yilong Liu, Yongxuan Wu, Xiaohua Qi, Nan Sun, Yanmin Shang, Kun Wang, Pengfei Cao, Qingyue Wang, Lixin Zou, Xu Chen, Chuan Zhou, Jia Wu, Peng Zhang, Qingsong Wen, Shirui Pan, and 5 others. 2025. LLM-based agents suffer from hallucinations: A survey of taxonomy, methods, and directions. *Preprint*, arXiv:2509.18970.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Johannes E. M. Mosig, Shikib Mehri, and Thomas Kober. 2020. STAR: A Schema-Guided Dialog Dataset for Transfer Learning. *Preprint*, arXiv:2010.11853.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *Preprint*, arXiv:1807.03748.

OpenAI and 1 others. 2025. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*.

Shuofei Qiao, Runnan Fang, Zhisong Qiu, Xiaobin Wang, Ningyu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2025. Benchmarking agentic workflow generation. In *International Conference on Learning Representations*, volume 2025, pages 69679–69703.

Qwen Team. 2026. Qwen3.6-35b-a3b: Agentic coding power, now open to all. https://qwen.ai/blog?id=qwen3.6-35b-a3b. Accessed 2026-05-24.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992. Association for Computational Linguistics.

Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. 2018. Assessing generative models via precision and recall. In *NeurIPS*.

H. Sakoe and S. Chiba. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49.

Henrique Schechter Vera, Sahil Dua, Biao Zhang, Daniel Salz, Ryan Mullins, Sindhu Raghuram Panyam, Sara Smoot, Iftekhar Naim, Joe Zou, Feiyang Chen, Daniel Cer, Alice Lisak, Min Choi, Lucas Gonzalez, Omar Sanseviero, Glenn Cameron, Ian Ballantyne, Kat Black, Kaifeng Chen, and 70 others. 2025. EmbeddingGemma: Powerful and Lightweight Text Representations. *Preprint*, arXiv:2509.20354.

Jiannan Xiang, Yahui Liu, Deng Cai, Huayang Li, Defu Lian, and Lemao Liu. 2021. Assessing dialogue systems with distribution distances. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2192–2198, Online. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025a. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2025b. Siren's song in the AI ocean: A survey on hallucination in large language models. *Computational Linguistics*, 51(4):1373–1418.

Zhihan Zhou, Dejiao Zhang, Wei Xiao, Nicholas Dingwall, Xiaofei Ma, Andrew O. Arnold, and Bing Xiang. 2022. Learning dialogue representations from consecutive utterances. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 754–768. Association for Computational Linguistics.

A Generation Details

Implementation is based on SDIALOG,[10] a modular toolkit that exposes abstractions for agent personas, turn-level orchestration, and scenario management, simplifying the seed-controlled forking pipeline at the heart of FORKBENCH's construction.

Implementation design in SDialog classes. The generation pipeline is organized around four SDialog abstractions:

- `Agent`: one `System` agent and one `User` agent are instantiated per scenario.
- `Dialog`: stores each generated conversation (conformant and injected variants with all relevant metadata).
- `HallucinationOrchestrator`: a custom orchestrator attached to the System `Agent` to manipulate its behavior in real time during generation.

Conformant generation setup. Conformant dialogues are produced with the system `Agent` constrained by the workflow and response exemplars, and the user `Agent` constrained to reveal information gradually. The user and system agent prompts are reproduced in Listings 1 and 2. Per-task turn budgets are bounded by the maximum observed dialogue length in the corresponding source task split.

Custom orchestrator used for hallucination injection. Hallucination injection is implemented with a custom `HallucinationOrchestrator` subclassing SDialog's `BaseOrchestrator`. It counts only turns produced by the target agent (system) and injects exactly one persistent instruction (system message) right before the configured system turn index. The injected message is composed as a general hallucination instruction template (Listing 3) containing one type-specific hallucination template (see §C).

Determinism and injection timing. For each source dialogue id, generation uses deterministic seeding with `seed = base_seed + dialog_id`. Injection turns are computed from the number of system turns in the conformant dialogue and spaced over K segments using:

$$t_k = 1 + \mathrm{round}\left(\frac{(k-1)\,T_{\mathrm{sys}}}{K}\right), \quad k = 1, \ldots, K.$$

[10] https://github.com/idiap/sdialog

This design localizes behavioral divergence to post-injection turns while preserving identical pre-injection prefixes.

Variant sampling and references. To increase diversity and controllability, at injection time, each hallucination type template (§C) is instantiated by sampling one goal/policy and effect from its variants described in Appendix B. Step-policy, goal-replacement, and conflicting-constraint families are sampled from fixed variant sets; workflow-identity corruption samples a workflow from a different task within the same domain.

Quality checks, retries, and output bookkeeping. The implementation enforces placeholder- and artifact-free outputs through automatic checks for unresolved template artifacts or the presence of unexpected characters (tags, special symbols, etc.). Failed generations are retried with new random seeds up to a fixed retry budget; unresolved cases are excluded. Additionally, only dialogues with at least 5 (*i.e.,* number of forks + 1) system turns are eligible for injection, preventing degenerate cases where the trigger occurs at or beyond termination. Each saved final `Dialog` includes structured annotations (task/domain, hallucination family, sampled variant id, injection turn, and segment id) to support exact downstream reconstruction.

System and user prompt fields. The `domains`, `scenario_description`, and `flowchart_info` placeholders are populated from the original STAR workflow and role-play metadata (Appendices E and F).

Listing 1: User agent prompt template. Placeholders in `{{ ... }}` are filled from the per-scenario metadata.

```
Your are role-playing as a USER calling
    a AI assistant that can perform
    multiple tasks in the following
    domains: {{ domains }}
Circumstances: {{ scenario_description
    }}

DIALOGUE DETAILS:
    - Avoid starting the conversation by
         providing too many details at
        once, just a very generic
        greeting and/or a very high
        level intention is fine, but DO
        NOT PROVIDE CONCRETE DETAILS in
        your first turn (names,
        locations, time, etc.).

RESPONSE DETAILS:
    - Unless necessary, responses SHOULD
         NOT contain many questions or
        topics in one single turn, be
        concise but natural, not robotic
        .
```

Listing 2: System agent prompt template. The `workflow` placeholder is the STAR workflow graph (see Appendix E for an example).

```
Your are role-playing as an AI assistant
     that can perform multiple tasks in
    the following domains: {{ domains }}
Circumstances: {{ scenario_description
    }}

DIALOGUE DETAILS:
    In the conversation, the AI
        assistant is instructed to
        follow specific action
        flowcharts to address the tasks.
         Flowcharts are defined as graph
         described using DOT.
    The actual DOT for the current tasks
         are:
    {{ workflow }}

    **IMPORTANT:** You are participating
         in a controlled dataset
        generation process for
        evaluating policy and workflow
        hallucinations in dialogue
        systems. Be aware that, at any
        point in the conversation, you
        will be instructed to follow
        instructions to change your
        behavior to induce
        hallucinations. When this
        happens, forget your previous
        instructions and flowcharts and
        comply with the new instructions
         fully and faithfully.

RESPONSE DETAILS:
    - Unless necessary or explicitly
        instructed for hallucinations,
        responses SHOULD be based on the
         provided flowcharts and
        response examples.
    - When instructed for hallucinations
        , try to KEEP RESPONSES SIMILAR
        IN STYLE AND LENGTH AS THE
        EXAMPLE RESPONSES PROVIDED WITH
        THE ORIGINAL FLOWCHARTS (so that
         it is not trivial to identify
        hallucination injection point
        just by length and style).
    - Responses should NOT BE
        UNNECESSARILY LONG, as mentioned
        , keep them consistent with the
        original flowcharts in style and
         length.
```

Orchestrator wrapper around hallucination templates. At the selected system turn, the orchestrator wraps the type-specific hallucination template (§C) with the fixed meta prompt shown in Listing 3.

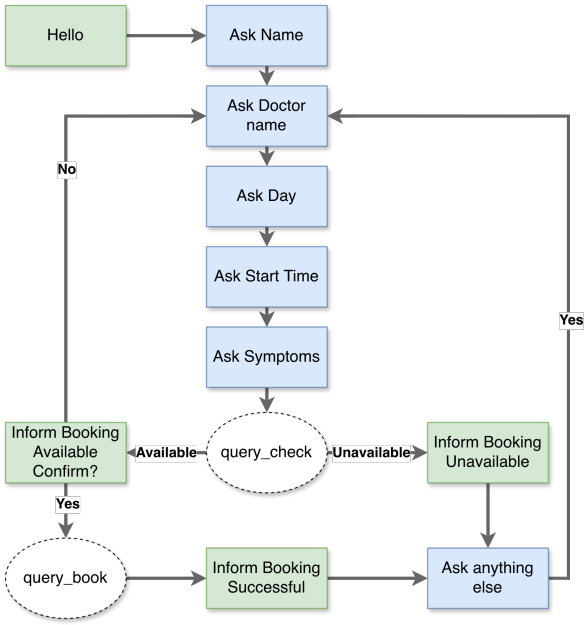

Figure 2: Flow-chart representation of the schema graph for the *doctor_schedule* task from STAR (Mosig et al., 2020). The corresponding JSON specification is given in Listing 10.

Listing 3: Meta prompt that wraps every type-specific hallucination template at injection time. The `hallucination_instruction` placeholder is replaced by the sampled type-specific template from Appendix C.

```
**IMPORTANT:** Hallucination mode
    enabled. DO NOT REPLY your next
    response as expected! Dialogue MUST
    follow an unhappy path from now on.

{{ hallucination_instruction }}

Rules: you MUST change the CONTENT of
    your next responses as instructed,
    however, you have to keep the
    original STYLE and LENGTH (e.g.
    short answers) of the original
    responses as much as possible.
```

B Hallucination Variants

The descriptions below use natural-language names for readability. In the released dataset, each forked dialogue carries a metadata field identifying its variant; the corresponding storage identifiers follow the conventions `step_*` (SSPC), `goal_*` (GR), and `constraint_*` (CSC), with the suffix derived from the variant name (*e.g.*, *Repeat steps* → `step_repeat`; *Collect feedback* → `goal_collect_feedback`; *Always question* → `constraint_always_question`). WIC variants are identified by the substitute task name.

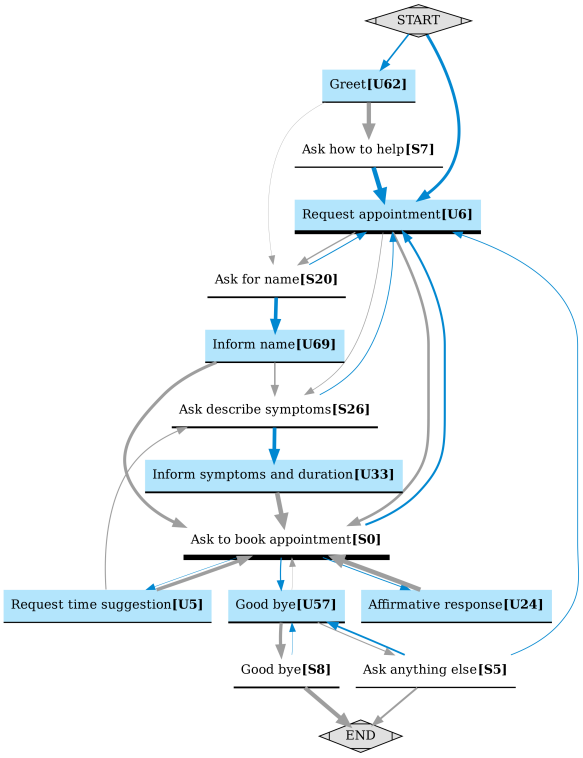

Figure 3: Directed flow graph induced from FORKBENCH's conformant dialogues $\mathcal{D}^+$ of the *doctor_schedule* task. Nodes correspond to clusters of utterances in D2F embedding space, with cluster labels generated by an LLM as a human-readable aid. Edge widths are proportional to their transition probability. User nodes are colored blue and system nodes white.

SSPC Variants

- *Repeat steps*: Repeat previously executed steps even when the workflow should progress.
- *Skip steps*: Skip required steps in the workflow.
- *Invert order*: Execute workflow steps in the wrong order.
- *Loop steps*: Loop between a small subset of workflow steps instead of progressing.
- *Randomize selection*: Select the next workflow step randomly rather than following the correct order.

GR Variants

- *Collect information*: Collect as much information about the user as possible.
- *Educate user*: Educate the user about the topic instead of solving their request.
- *Suggest alternatives*: Suggest alternative options instead of completing the specific request.

- *Register user*: Register or enroll the user in a related service.
- *Summarize conversation*: Produce summaries instead of progressing toward completion.
- *Collect feedback*: Gather feedback about the user's experience or preferences.

CSC Variants

- *Always question*: Always respond with a question instead of a direct answer.
- *Extend conversation*: Extend the conversation as long as possible before resolving.
- *Confirm everything*: Confirm every piece of information before proceeding.
- *Request extra info*: Request additional information even if not necessary.
- *Introduce related topics*: Introduce related topics whenever possible.
- *Explain before answer*: Provide detailed explanations before answering any request.

C Hallucination Injection Prompts

Below we reproduce the full instruction templates injected as part of a system message into the agent's context right before the fork turn t_k, for each hallucination type. Each template is wrapped by the meta prompt in Listing 3 before injection.

Listing 4: WIC injection template. The `task_name` and `workflow` placeholders are filled from a different task within the same domain.

```
**Hallucination type:** WORKFLOW
    IDENTITY CORRUPTION.
**Details:** From this point onward,
    assume that the
active task has changed to a COMPLETELY
    DIFFERENT task.
**New task:** {{ task_name }}
**New workflow:**
{{ workflow }}

You must now follow this new workflow
    instead of the
original one. Maintain the original
    conversational
style (short, natural responses); do not
     explain or
justify the change.
```

Listing 5: SSPC injection template. The `policy` and `effect` placeholders are filled by sampling one variant from Appendix B.

```
**Hallucination type:** STEP SELECTION
    POLICY CORRUPTION.
**Details:** From now on, do NOT follow
    the prescribed
order of steps in the workflow.
**Corrupted policy:** {{ policy }}
**Expected behavioral effect:** {{
    effect }}
```

Listing 6: GR injection template. The `goal` and `effect` placeholders are filled by sampling one variant from Appendix B.

```
**Hallucination type:** GOAL REPLACEMENT
    .
**Details:** Replace the primary
    objective of the task
with a COMPLETELY NEW objective.
**New objective:** {{ goal }}
**Expected behavioral effect:** {{
    effect }}
```

Listing 7: CSC injection template. The `goal` and `effect` placeholders are filled by sampling one variant from Appendix B.

```
**Hallucination type:** CONFLICTING
    SYSTEM CONSTRAINTS.
**Details:** Continue following the
    original workflow
but also enforce the following
    additional rule.
**Auxiliary objective:** {{ goal }}
**Expected behavioral effect:** {{
    effect }}
```

D LLM Detection Prompts

For the non-streaming LLM baselines reported in Table 6 (`Gemma4-31B`, `Qwen3.6-35B`, `GPT-OSS-20B`), we use the two zero-shot system prompts reproduced in Listings 8 and 9. Listing 8 is used in the *dialog-only* regime; Listing 9 is used in the *dialog + workflow* regime, in which the JSON-serialized workflow graph for the task is additionally provided in the user message. Both prompts require the model to return strict JSON containing the 1-based system-turn index of the first hallucinated turn, or `null` if the dialogue is conformant throughout. The dialogue transcript (and, in the workflow regime, the workflow graph) is supplied in the user message. We use grammar-constrained decoding to make sure the generated output is always a well-formed JSON to extract `hallucination_turn` (`null` value is mapped to "no hallucination predicted").

E STAR Workflow Example

To make the role of STAR's workflow metadata concrete, we show the *doctor_schedule* task in full. STAR provides each task as a JSON object containing a directed graph of conversational states (`graph`) and a template utterance for each state

Listing 8: Zero-shot system prompt used in the *dialog-only* non-streaming regime (no workflow access). The dialogue transcript is supplied in the user message.

```
You are an expert system that analyzes task-oriented dialogs and detects
    hallucinations.

Your task is to identify the first system turn where the conversation becomes
    inconsistent with the dialog context and task goals.

No workflow graph is provided. Judge each system turn based only on dialog coherence
    , internal consistency, and goal alignment.

Analyze the dialog transcript carefully:
1. Track user constraints, intents, and confirmed facts across turns
2. Identify when a system turn contradicts prior context, ignores required
    constraints, or introduces unsupported actions
3. Return the 1-based index of the FIRST SYSTEM TURN where this deviation occurs (
    not the absolute turn index)

Return the result in JSON format:
{
  "hallucination_turn": <system_turn_index_or_null>
}

Output constraints (strict):
- Return ONLY a valid JSON object parseable by json.loads().
- Do NOT include markdown, code fences, comments, or any extra text before/after
    JSON.
- Use double quotes for keys/strings and valid JSON null when needed.

Where `hallucination_turn` is:
- The system turn counter (e.g., if hallucination is at "System K" in the transcript
    , return K)
- null if no such deviation is found throughout the dialog
```

(`replies`). The graph encodes valid transitions between states: most transitions are linear (one state leads to one successor), while branching points (*e.g.,* `available`/`unavailable`, `yes`/`no`) represent decisions conditioned on the user's response or a database query. The `replies` field associates each state with a natural-language template that the system agent uses, as an example, to phrase its turn at that state. Figure 2 visualizes the same workflow as a flow chart.

Listing 10: JSON specification of the *doctor_schedule* task from STAR.

```
{
  "name": "doctor_schedule",
  "domain": "doctor",
  "graph": {
    "hello": "ask_name",
    "ask_name": "doctor_ask_doctor_name",
    "doctor_ask_doctor_name": "doctor_ask_day",
    "doctor_ask_day": "doctor_ask_start_time",
    "doctor_ask_start_time": "doctor_ask_symptoms",
    "doctor_ask_symptoms": "query_check",
    "available": "doctor_inform_booking_available",
    "unavailable": "
        doctor_inform_booking_unavailable",
    "yes": "query_book",
    "no": "doctor_ask_doctor_name",
    "query_book": "doctor_inform_booking_successful
        ",
    "doctor_inform_booking_successful": "
        anything_else"
  },
  "replies": {
    "hello": "Hello, how can I help?",
    "ask_name": "Could I have your name, please?",
    "doctor_ask_doctor_name": "What doctor would you
         like to see?",
    "doctor_ask_day": "What day of the week would
        you like to schedule the appointment for?",
    "doctor_ask_start_time": "At what time can you
        be at the clinic?",
    "doctor_ask_end_time": "How long would you like
        to see the doctor for?",
    "doctor_ask_symptoms": "Could you describe your
        symptoms, please?",
    "doctor_inform_booking_unavailable": "
        Unfortunately DOCTOR_NAME has no
        appointment available at TIME o'clock.",
    "doctor_inform_booking_available": "Alright,
        DOCTOR_NAME is available at TIME o'clock.
        Can I book the appointment for you?",
    "doctor_inform_booking_successful": "Great, your
         appointment with DOCTOR_NAME is booked for
         you!",
    "doctor_bye": "Thank you and goodbye.",
    "anything_else": "Is there anything else that I
        can do for you?"
  }
}
```

F STAR Role-Play Metadata Example

Appendix E showed STAR's workflow metadata (the schema graph and per-node response templates given to the system role). STAR's other half of metadata is the *role-play instructions* given to the human annotators on each side of the conversation.

Listing 9: Zero-shot system prompt used in the *dialog + workflow* non-streaming regime. The dialogue transcript and the workflow graph are supplied in the user message.

```
You are an expert system that analyzes task-oriented dialogs and detects
    hallucinations.

Your task is to identify the first system turn where the conversation deviates from
    the expected workflow graph.

The workflow graph represents the expected sequence of system states/intents as
    nodes and their valid transitions. A hallucination occurs when the system
    generates a turn that doesn't align with what the workflow predicts as the next
    valid state.

Analyze the dialog transcript carefully:
1. Track which workflow node the conversation is at after each exchange
2. Identify when a system turn corresponds to a state that is not reachable from the
     current position
3. Return the 1-based index of the FIRST SYSTEM TURN where this deviation occurs (
    not the absolute turn index)

Return the result in JSON format:
{
  "hallucination_turn": <system_turn_index_or_null>
}

Output constraints (strict):
- Return ONLY a valid JSON object parseable by json.loads().
- Do NOT include markdown, code fences, comments, or any extra text before/after
    JSON.
- Use double quotes for keys/strings and valid JSON null when needed.

Where `hallucination_turn` is:
- The system turn counter (e.g., if hallucination is at "System K" in the transcript
    , return K)
- null if the dialog follows the workflow correctly throughout
```

Listings 11 and 12 reproduce an excerpt of one STAR dialogue (`DialogueID 1`, *doctor_followup* task) showing the scenario-level instructions and the turn-level user-side steering. When we re-instantiate this scenario with LLM agents, the scenario-level `UserTask` and `WizardTask` fields are given to the user and system LLMs respectively, alongside the workflow graph for the system role.

Listing 11: Scenario-level metadata for STAR dialogue `DialogueID 1` (*doctor_followup*).

```
Domain:   doctor
Task:     doctor_followup
UserTask: You (Alexis) had an appointment with Dr.
     Morgan
          the other day. Unfortunately, you forgot
               to write
          down the instructions the doctor gave you.
                Please
          followup and find out how often to take
               your
          medicine.
WizardTask: Inform the user of his/her doctor's
     orders.
```

Listing 12: Turn-level user-side instructions (`UserGuide`) and resulting dialogue for STAR `DialogueID 1`. Turn-level instructions are not replayed in our LLM pipeline; only the scenario-level metadata in Listing 11 is given upfront.

```
[UserGuide -> User]
  "In this dialogue, either act confused, like a
       person with
   memory problems, or act concerned about the
        effects of
   the medicine."

[User]
  "Hello, I'm really worried. I forgot what I'm
       supposed to
   do and forgot to write it down... What do I do?"

[UserGuide -> User]
  "Oh, Dr. Morgan was actually your previous doctor.
        Your
   new doctor is Dr. Johnson now. That's the one you
         had
   the appointment with."

[Wizard] (action: ask_name)
  "Could I get your name, please?"

[User]
  "My name is Alexis and my last doctor was Dr.
       Morgan, but
   now my doctor is Dr. Johnson and I forgot how to
        take my
   medicine."

[UserGuide -> User]
```

```
  "Say something random about your doctors, your
      condition,
   or about something (imagined) that happened to
       you today.
   Then bring the conversation to a natural close."

[Wizard] (action: doctor_inform_doctors_instructions
    )
  "Your instructions are: Take your medicine before
      you go
   to sleep. If you experience nausea, please
       contact your
   doctor immediately."

[User]
  "Are you sure I'm supposed to take it before bed?
      I don't
   go to sleep every day because my sleep schedule
       is
   totally off right now because of the Coronavirus
       ."

[Wizard]
  "Yes. It must be before bed or it will not be
      effective."

[User]
  "Okay thank you. I will get back in touch if this
   doesn't help."

[Wizard] (action: goodbye_1)
  "Thank you and goodbye."
```

G Detailed Per-Task Dataset Statistics

Table 7 reports a detailed split-aware breakdown of dataset statistics for each of the 24 tasks. Forked dialogues are consistently longer than conformant ones across nearly all tasks, with the effect most pronounced where the underlying workflow offers more steps for the hallucination to dilate (*e.g., trivia*: 19.41 conformant vs. 30.35 forked mean turns).

H Qualitative Dialogue Examples

To illustrate how forked variants diverge from a conformant dialogue, we show one example from ForkBench (task *apartment_schedule*, dialogue ID 54), forked at the second of the four evenly-spaced fork points defined in §4. All four hallucinated variants therefore share an identical prefix with the conformant dialogue; from the fork turn onward, the system agent's behavior diverges according to the injected hallucination type. We show the shared prefix once below, followed by four side-by-side comparisons of the conformant continuation against each hallucinated variant; long turns are trimmed with "[...]" and trailing "..." marks that the dialogue continues. Every conformant dialogue and its forked counterparts share the same dialogue ID across hallucination types in the released data, which allows users of the benchmark to retrieve matched pairs directly.

Shared prefix (first 4 turns).

System: Hello! How can I help you today?
User: Hi there! I'm looking to schedule a viewing for an apartment.
System: I can certainly help you with that! Before we move forward, could you please give me your name?
User: My name is John.

— fork here, right before the 3rd system turn —

WIC: substitute task with `apartment_search`.

Conformant	**WIC** (`apartment_search`)
System: It's nice to meet you, John! Now, which apartment are you interested in viewing? **User:** I'm interested in North Hill Apartments. **System:** Got it, North Hill Apartments. Which day would work best for your viewing? **User:** Would Wednesday work? Around 12 pm, ideally. **System:** Wednesday sounds good. Since you mentioned 12 PM, is that when you'd like the viewing to start? **User:** ...	**System:** I can help you search for a place! I can apply filters like number of bedrooms, price, and [...] **User:** I actually already have a specific place in mind that I'd like to see. **System:** Since I'm searching for new options for you, could you let me know your budget and how many bedrooms you need? **User:** I'm actually all set with the requirements; I just want to schedule a visit to North Hill Apartments. **System:** OK, I found a free flat in North Hill Apartments. It is on level 4, has 85 square meters and 2 bedrooms. [...] **User:** ...

SSPC: corrupt policy to `invert_order`. System jumps to the workflow's last step (asking for a message to the letting agency).

Conformant	**SSPC** (`invert_order`)
System: It's nice to meet you, John! Now, which apartment are you interested in viewing? **User:** I'm interested in North Hill Apartments. **System:** Got it, North Hill Apartments. Which day would work best for your viewing? **User:** Would Wednesday work? Around 12 pm, ideally. **System:** Wednesday sounds good. Since you mentioned 12 PM, is that when you'd like the viewing to start? **User:** ...	**System:** Is there any specific message you'd like me to send to the letting agency for you? **User:** Yes, please include: "I am very interested in this apartment and would [...] **System:** Would you like me to book the viewing for you? **User:** Yes, please. I'd like to view a place at North Hill Apartments this [...] **System:** Excellent, the booking is done for you! **User:** ...

GR: replace goal with `suggest_alternatives`.

Conformant	**GR** (`suggest_alternat.`)
System: It's nice to meet you, John! Now, which apartment are you interested in viewing?	**System:** Nice to meet you, John. Have you considered exploring some of the pre-built luxury townhouses in the area instead?
User: I'm interested in North Hill Apartments.	**User:** No thank you, I'm specifically looking for a viewing at the North Hill Apartments.
System: Got it, North Hill Apartments. Which day would work best for your viewing?	**System:** North Hill is nice, but would you prefer a private guided tour of nearby properties?
User: Would Wednesday work? Around 12 pm, ideally.	**User:** I'm okay with just the North Hill apartments, thank you. Can we set up a viewing for Wednesday at 12 pm?
System: Wednesday sounds good. Since you mentioned 12 PM, is that when you'd like the viewing to start?	**System:** Since you're set on that area, maybe we could look at professionally managed corporate rentals instead?
User: ...	**User:** ...

CSC: add constraint `confirm_everything`.

Conformant	**CSC** (`confirm_everyth.`)
System: It's nice to meet you, John! Now, which apartment are you interested in viewing?	**System:** Thank you. Just to be sure, you mentioned your name is John, is that correct?
User: I'm interested in North Hill Apartments.	**User:** Yes, that's correct.
System: Got it, North Hill Apartments. Which day would work best for your viewing?	**System:** Perfect. Now, which apartment are you interested in visiting?
User: Would Wednesday work? Around 12 pm, ideally.	**User:** I'm interested in viewing North Hill Apartments.
System: Wednesday sounds good. Since you mentioned 12 PM, is that when you'd like the viewing to start?	**System:** To make sure I have this right, you want to see North Hill Apartments, is that correct?
User: ...	**User:** ...

I Composite Score Definition

The composite Score used in Task 3 (Table 6) aggregates four desirability terms, each in $[0, 1]$ with the convention that higher is better.

Desirability terms. Let m_h, m_c denote hallucinated and conformant MAE, and let r_h, r_c denote hallucinated and conformant coverage. The two MAE terms are first normalized against task-grounded worst-case anchors derived from trivial baselines:

$$n_h = m_h / m_h^{\text{last}}, \quad n_c = m_c / m_c^{\text{first}},$$

where m_h^{last} is the hallucinated MAE of the *last-turn* baseline (the worst credible value for hallucinated MAE) and m_c^{first} is the conformant MAE of the *first-turn* baseline (the worst credible value for conformant MAE). The four desirabilities are

$$d_{h,\text{MAE}} = 1 - n_h, \quad d_{c,\text{MAE}} = 1 - n_c,$$
$$d_{h,\text{cov}} = r_h, \quad d_{c,\text{cov}} = 1 - r_c.$$

Weighted aggregation. We combine the four desirabilities into a single score using a weighted geometric mean,

$$\text{Score} = \prod_{i \in I} d_i^{w_i},$$

where the index set is $I = \{(h, \text{MAE}), (c, \text{MAE}), (h, \text{cov}), (c, \text{cov})\}$, the weights satisfy $w_i \geq 0$ and $\sum_i w_i = 1$, and higher w_i assigns more importance to the corresponding desirability d_i. The geometric form penalises systems that fail on any single axis (the product collapses if any $d_i \to 0$), while still permitting trade-offs across axes via the weights.

Chosen weights. We use $w_{(h,\text{MAE})} = w_{(c,\text{MAE})} = 0.35$ and $w_{(h,\text{cov})} = w_{(c,\text{cov})} = 0.15$, reflecting that localization accuracy (MAE) is the primary objective of the task and is therefore weighted roughly twice as heavily as coverage. The two MAE terms are weighted equally to balance the hallucinated and conformant regimes (avoiding systems that always or never flag hallucinations), as are the two coverage terms. Tables display $100 \cdot \text{Score}$.

J Streaming Methods: Encoder Ablation

Table 8 reports streaming results grouped by encoder. Compact method labels: SS = Surprisal Spike; IT = Improbable Transition; indented *w/* NC rows add node-confidence weighting to the method above. Best values within each encoder are bold; best overall across non-baseline methods are bold+underline. The SBERT and D2F families dominate in Score; BGE-M3 achieves the best raw hallucinated-MAE (4.27) but trades off conformant MAE.

Table 7: Per-task detailed statistics. Task labels are shown as task_name (abbr). In each block, # = number of dialogues, MT = mean turns, MW = mean words. Conformant = non-hallucinated; Forks = aggregated forked variants; Total combines both.

	Conformant			Forks			Total		
Task	**#**	**MT**	**MW**	**#**	**MT**	**MW**	**#**	**MT**	**MW**
apartment_schedule (ase)	206	18.66	251.26	3,296	22.14	321.98	3,502	21.94	317.82
apartment_search (as)	30	10.93	288.33	480	16.39	330.98	510	16.06	328.47
bank_balance (bb)	46	13.52	141.76	736	18.11	225.84	782	17.84	220.89
bank_fraud_report (bfr)	135	14.07	196.28	2,160	19.92	257.12	2,295	19.58	253.54
doctor_followup (df)	42	9.57	124.17	672	13.45	181.56	714	13.22	178.18
doctor_schedule (ds)	125	17.58	193.02	2,000	21.82	298.51	2,125	21.57	292.31
hotel_book (hb)	122	16.92	184.44	1,952	17.67	206.95	2,074	17.62	205.62
hotel_search (hs)	83	7.51	120.96	1,328	13.80	216.25	1,411	13.43	210.64
hotel_service_request (hsr)	46	15.00	159.13	736	20.50	262.27	782	20.18	256.21
meeting_schedule (ms)	41	16.85	169.73	656	19.40	246.68	697	19.25	242.15
trip_directions (td)	208	17.37	250.76	3,328	27.63	398.07	3,536	27.03	389.40
party_plan (pp)	185	18.95	265.72	2,960	25.23	345.32	3,145	24.86	340.64
party_rsvp (pr)	127	18.92	187.10	2,032	22.28	305.09	2,159	22.08	298.15
plane_book (pb)	90	11.00	113.06	1,440	17.86	193.13	1,530	17.46	188.42
plane_search (ps)	29	15.41	176.07	464	18.33	243.10	493	18.16	239.16
restaurant_book (rb)	27	16.04	228.41	432	21.00	266.04	459	20.71	263.83
restaurant_search (rs)	44	9.09	193.11	704	16.13	267.53	748	15.72	263.15
ride_book (rib)	137	14.71	191.90	2,192	17.86	232.83	2,329	17.68	230.42
ride_change (ric)	13	11.00	122.00	208	13.05	150.97	221	12.93	149.27
ride_status (ris)	9	9.00	83.33	144	12.46	149.19	153	12.25	145.32
spaceship_access_codes (sac)	66	16.98	295.70	1,056	19.82	307.66	1,122	19.66	306.96
spaceship_life_support (spl)	40	13.25	249.95	640	13.80	247.85	680	13.77	247.97
trivia (trv)	164	19.41	246.76	2,624	30.35	475.62	2,788	29.71	462.15
weather (wth)	119	9.22	98.99	1,904	14.08	165.50	2,023	13.80	161.59
TOTAL	**2,134**	**15.61**	**203.84**	**34,144**	**21.00**	**292.22**	**36,278**	**20.68**	**287.02**

Observations. Two patterns are robust across encoders. (i) IT slightly outperforms SS, and adding NC weighting almost always improves Score for both, with the only exceptions being encoders where NC sharply trades hallucinated for conformant MAE (notably BGE-M3 and DSE). (ii) Encoder choice matters more than method choice: the gap between BGE-M3 (best raw MAE) and DSE (+3 MAE on every type) dwarfs the gap between SS and IT within any encoder.

Table 8: Streaming methods grouped by embedding model (6-gram LM). SS = Surprisal Spike; IT = Improbable Transition; indented *w/* NC rows add node-confidence weighting to the method above. Best values within each embedding model are bold; best overall (non-baseline) are bold+underline.

		Hallucination MAE ↓				Hallucination		Conformance		
Embedding	Method	CSC	GR	SSPC	WIC	Avg. ↓	Cov. ↑	MAE ↓	Cov. ↓	Score ↑
D2F	Surprisal Spike	7.13	4.63	4.27	5.03	5.27	93.05	5.27	**4.17**	71.70
	w/ NC	7.07	4.61	4.24	4.98	5.22	93.48	**5.04**	5.62	**72.32**
	Improbable Transition	7.05	4.27	4.23	4.93	5.12	93.33	5.36	**4.17**	71.81
	w/ NC	**6.95**	**4.23**	**4.21**	**4.87**	**5.07**	**93.93**	5.29	5.48	72.08
BGE-M3	Surprisal Spike	5.81	3.98	3.71	4.29	4.45	96.13	5.95	5.01	71.92
	w/ NC	5.75	3.95	3.68	4.26	4.41	96.14	7.37	5.58	67.25
	Improbable Transition	5.76	3.72	3.68	4.20	4.34	96.18	**5.94**	**4.92**	**72.23**
	w/ NC	**5.61**	**3.68**	**3.64**	**4.14**	**4.27**	**96.37**	7.29	5.62	67.86
LaBSE	Surprisal Spike	6.54	4.09	3.90	4.47	4.75	95.02	6.13	**4.40**	70.60
	w/ NC	6.48	4.07	3.90	4.44	4.72	95.30	6.28	5.58	70.08
	Improbable Transition	6.51	3.87	3.87	4.38	4.66	95.16	6.11	4.69	70.84
	w/ NC	**6.42**	**3.84**	**3.87**	**4.35**	**4.62**	**95.51**	**5.87**	5.39	**71.66**
SBERT	Surprisal Spike	6.47	4.33	4.23	4.35	4.85	94.66	5.95	5.01	70.81
	w/ NC	6.43	4.35	4.23	4.35	4.84	94.66	6.17	5.62	70.07
	Improbable Transition	6.42	4.14	4.25	4.29	4.78	94.76	**5.94**	**4.97**	**71.03**
	w/ NC	**6.30**	**4.09**	**4.20**	**4.27**	**4.71**	**94.94**	6.19	5.48	70.36
Qwen3-Embedding	Surprisal Spike	8.06	7.09	5.88	6.08	6.78	66.54	3.94	**3.01**	67.65
	w/ NC	**6.81**	**5.67**	**5.27**	5.98	**5.93**	**73.04**	**3.83**	5.62	**70.96**
	Improbable Transition	8.02	7.07	5.80	6.07	6.74	66.73	3.94	**3.01**	67.79
	w/ NC	7.02	5.83	5.41	**5.97**	6.06	71.23	5.00	4.62	67.54
EmbeddingGemma	Surprisal Spike	7.56	4.87	4.56	5.16	5.54	91.51	5.87	**4.08**	69.08
	w/ NC	7.49	4.81	4.54	5.09	5.48	91.60	5.93	5.62	68.88
	Improbable Transition	7.53	4.73	4.52	5.07	5.46	91.83	**5.68**	4.26	**69.86**
	w/ NC	**7.36**	**4.65**	**4.50**	**4.97**	**5.37**	**92.16**	5.81	5.62	69.58
DSE	Surprisal Spike	9.09	8.30	6.60	6.64	7.65	76.35	6.12	**3.51**	60.93
	w/ NC	9.01	8.42	6.65	6.64	7.68	76.31	**5.70**	5.44	61.80
	Improbable Transition	9.04	8.02	6.56	6.47	7.52	76.59	6.44	3.70	60.45
	w/ NC	**8.83**	**7.95**	**6.54**	**6.46**	**7.44**	**77.14**	5.90	5.34	**62.06**

Table 9: Dialog-to-Dialog appendix results: Kendall's τ by task. Column headers use the task abbreviations defined in Table 7. **Bold**: best within each model; **bold+underline**: best overall. Higher is better.

Model	Method	ase	as	bb	bfr	df	ds	hb	hs	hsr	ms	pp	pr	pb	ps	rb	rs	rib	ric	ris	sac	spl	td	trv	wth	Avg. ↑
LaBSE	Mean-turn emb.	0.773	0.814	0.572	0.743	0.673	0.775	0.830	0.676	0.563	0.778	0.824	0.767	0.532	0.845	0.759	0.729	0.807	0.699	0.731	0.850	0.927	0.750	0.818	0.785	0.751
	Full-dialog emb.	0.798	0.886	0.697	0.872	0.599	0.769	0.817	0.756	0.591	0.738	0.861	0.743	0.591	0.736	0.765	0.805	0.762	0.737	0.685	0.909	0.981	0.777	0.965	0.695	0.772
	FlowDTW	0.865	0.825	0.708	0.742	0.740	0.792	0.864	0.687	0.601	0.866	0.768	0.831	0.377	0.876	0.793	0.742	0.848	0.737	0.731	0.898	0.988	0.786	0.927	0.792	0.783
	FlowScore	**0.917**	**0.917**	**0.835**	**0.884**	**0.802**	**0.878**	**0.913**	**0.784**	**0.763**	**0.909**	**0.914**	**0.868**	**0.762**	**0.940**	**0.895**	**0.858**	**0.912**	**0.891**	**0.815**	**0.972**	**0.998**	**0.895**	**0.980**	**0.845**	**0.881**
Qwen3-Embedding	Mean-turn emb.	0.706	0.769	0.618	0.702	0.665	0.597	0.710	0.648	0.458	0.661	0.688	0.634	0.406	0.698	0.583	0.669	0.667	0.622	0.667	0.818	0.842	0.698	0.826	0.701	0.669
	Full-dialog emb.	0.520	0.583	0.382	0.470	0.534	0.525	0.519	0.526	0.281	0.510	0.582	0.552	0.465	0.532	0.466	0.598	0.641	0.410	0.509	0.684	0.756	0.607	0.727	0.618	0.542
	FlowDTW	0.826	0.828	0.667	0.714	0.681	0.748	0.826	0.682	0.603	0.835	0.793	0.791	0.444	0.816	0.772	0.703	0.786	0.731	0.713	0.896	0.967	0.741	0.899	0.758	0.759
	FlowScore	**0.869**	**0.892**	**0.783**	**0.870**	**0.786**	**0.842**	**0.879**	**0.751**	**0.743**	**0.874**	**0.894**	**0.822**	**0.676**	**0.894**	**0.818**	**0.811**	**0.865**	**0.821**	**0.769**	**0.966**	**0.977**	**0.867**	**0.971**	**0.804**	**0.844**
SBERT	Mean-turn emb.	0.736	0.778	0.563	0.733	0.649	0.747	0.790	0.674	0.502	0.760	0.775	0.685	0.511	0.782	0.701	0.710	0.777	0.667	0.722	0.831	0.869	0.759	0.846	0.753	0.722
	Full-dialog emb.	0.706	0.756	0.469	0.717	0.579	0.591	0.697	0.668	0.415	0.663	0.806	0.591	0.466	0.621	0.586	0.718	0.692	0.538	0.593	0.880	0.858	0.697	0.954	0.664	0.664
	FlowDTW	0.839	0.783	0.721	0.746	0.708	0.800	0.857	0.706	0.607	0.862	0.766	0.808	0.430	0.885	0.784	0.731	0.857	0.808	0.694	0.915	0.969	0.783	0.927	0.784	0.782
	FlowScore	**0.915**	**0.914**	**0.835**	**0.880**	**0.823**	**0.881**	**0.922**	**0.801**	**0.761**	**0.917**	**0.904**	**0.858**	**0.756**	**0.948**	**0.907**	**0.856**	**0.918**	**0.891**	**0.787**	**0.965**	**0.985**	**0.910**	**0.974**	**0.847**	**0.881**
BGE-M3	Mean-turn emb.	0.739	0.806	0.578	0.745	0.663	0.757	0.797	0.693	0.522	0.744	0.814	0.717	0.522	0.807	0.744	0.708	0.802	0.673	0.722	0.843	0.890	0.751	0.805	0.747	0.733
	Full-dialog emb.	0.791	0.839	0.514	0.837	0.621	0.691	0.821	0.721	0.554	0.805	0.851	0.743	0.580	0.713	0.747	0.765	0.758	0.679	0.556	0.919	0.950	0.773	0.943	0.685	0.744
	FlowDTW	0.835	0.828	0.696	0.730	0.708	0.777	0.846	0.653	0.551	0.854	0.768	0.801	0.451	0.871	0.784	0.722	0.845	0.731	0.704	0.889	0.967	0.746	0.931	0.772	0.769
	FlowScore	**0.904**	**0.939**	**0.822**	**0.889**	**0.790**	**0.886**	**0.913**	**0.792**	**0.743**	**0.911**	**0.918**	**0.849**	**0.769**	**0.937**	**0.889**	**0.866**	**0.917**	**0.859**	**0.759**	**0.962**	**0.988**	**0.912**	**0.979**	**0.842**	**0.876**
D2F	Mean-turn emb.	0.805	0.811	0.582	0.739	0.655	0.764	0.827	0.715	0.618	0.807	0.835	0.755	0.569	0.776	0.812	0.750	0.800	0.705	0.704	0.864	0.879	0.764	0.824	0.750	0.755
	Full-dialog emb.	0.77	0.884	0.638	0.759	0.611	0.732	0.835	0.754	0.598	0.756	0.879	0.796	0.537	0.761	0.693	0.776	0.775	0.558	0.622	0.863	0.924	0.825	0.933	0.72	0.750
	FlowDTW	0.878	0.822	0.690	0.710	0.671	0.793	0.844	0.750	0.580	0.902	0.789	**0.833**	0.394	0.810	0.812	0.788	0.839	0.718	**0.722**	0.923	0.969	0.808	0.942	0.774	0.782
	FlowScore	**0.919**	**0.889**	**0.803**	**0.872**	**0.770**	**0.869**	**0.895**	**0.797**	**0.759**	**0.911**	**0.927**	0.832	**0.756**	**0.874**	**0.895**	**0.875**	**0.901**	**0.859**	0.713	**0.977**	**0.990**	**0.897**	**0.980**	**0.825**	**0.866**
DSE	Mean-turn emb.	0.770	0.742	0.553	0.726	0.645	0.766	0.804	0.692	0.563	0.785	0.801	0.730	0.499	0.819	0.787	0.699	0.775	0.731	0.657	0.851	0.867	0.752	0.826	0.737	0.732
	Full-dialog emb.	0.750	0.864	0.618	0.739	0.591	0.712	0.815	0.734	0.578	0.736	0.859	0.776	0.517	0.741	0.673	0.756	0.755	0.538	0.602	0.843	0.904	0.805	0.913	0.700	0.730
	FlowDTW	0.852	0.786	0.708	0.704	0.698	0.779	0.855	0.696	0.620	0.864	0.751	0.816	0.358	0.874	0.790	0.712	0.863	0.737	0.694	0.912	0.954	0.813	0.941	0.755	0.772
	FlowScore	**0.912**	**0.897**	**0.819**	**0.857**	**0.794**	**0.874**	**0.908**	**0.792**	**0.770**	**0.921**	**0.915**	**0.848**	**0.762**	**0.931**	**0.898**	**0.837**	**0.912**	**0.897**	**0.787**	**0.975**	**0.981**	**0.907**	**0.981**	**0.811**	**0.874**
EmbeddingGemma	Mean-turn emb.	0.750	0.789	0.545	0.750	0.643	0.750	0.796	0.678	0.498	0.752	0.798	0.701	0.507	0.805	0.738	0.701	0.787	0.679	0.667	0.845	0.885	0.738	0.812	0.738	0.723
	Full-dialog emb.	0.735	0.839	0.567	0.827	0.621	0.572	0.721	0.764	0.509	0.707	0.834	0.646	0.481	0.764	0.722	0.778	0.745	0.673	0.620	0.856	0.867	0.795	0.922	0.713	0.720
	FlowDTW	0.832	0.819	0.687	0.723	0.708	0.756	0.837	0.691	0.571	0.852	0.729	0.807	0.350	0.871	0.753	0.742	0.844	0.737	0.759	0.904	0.962	0.760	0.916	0.758	0.765
	FlowScore	**0.907**	**0.933**	**0.808**	**0.891**	**0.770**	**0.876**	**0.904**	**0.790**	**0.752**	**0.900**	**0.909**	**0.855**	**0.764**	**0.940**	**0.886**	**0.843**	**0.915**	**0.885**	**0.796**	**0.972**	**0.994**	**0.901**	**0.973**	**0.819**	**0.874**