Who Acts, to Whom, and What They Say: Action-Aware Social Simulation at Scale

Anonymous authors
Paper under double-blind review

Abstract

Behavioural foundation models answer what a given person would say: the speaker is an input. A population simulator must also decide who acts, when, and to whom, and today these decisions are typically left to hand-written rules. We pose full-event simulation, a task in which the speaker is an output: from a platform's history, a model emits the next events (how many, when, who, to whom, and what they say) and continues from its own output. We build ACT-AWARE, which learns who acts and whom they answer, and writes what they say with a post-trained behavioural foundation model. On a single-step version of the task, ACT-AWARE names real actors with 49.6% precision, against 36.6% for the best of fifteen language models prompted zero-shot. Rolled out the natural way, such a simulator meets a limit no model can lift. If it draws actors from a trailing window of recent posters and feeds on its own output, its coverage, the share of real actors it selects, has a ceiling fixed before its first step. A controller lets ACT-AWARE widen that window to seven days with no observed drop in precision. Over five held-out days, its coverage reaches $2.35\times$ the ceiling of a six-hour window, against $1.27\times$ for random ranking and $1.85\times$ for ranking by lifetime post count, both under the same controller and seven-day pool. Structurally, its output also becomes the world later events come from: 82.4% of its simulated replies attach to posts it generated, against 85.2% in real data. In short, who acts is a learnable factor rather than a scheduling convention.

1 Introduction

Agent-based social simulation built on language models is now a standard instrument for studying collective behaviour (Park et al., 2023; Yang et al., 2024; Piao et al., 2025; Tang et al., 2025; Wang et al., 2025; Zhang et al., 2025). These systems inherit one division of labour. The language model decides what an agent says, and a scheduler outside it decides who speaks and when. That scheduler is a rule: every agent acts each round (Park et al., 2023; Wang et al., 2025), each is activated from a per-hour rate fitted to its own history (Yang et al., 2024), or actors are sampled to match a target distribution (Zhang et al., 2025). A population is not a conversation. At any instant a tiny, time-varying subset of millions of accounts is about to act (Barabási, 2005; Perra et al., 2012), and which subset it is determines the macroscopic shape of everything downstream (Axtell, 2000). When the scheduler is a rule, the simulation's population dynamics are the rule's dynamics. Recent work keeps that rule and calibrates it (Zhang et al., 2026).

Full-event simulation (§2). We instead pose the task directly: from a platform's history, a model emits the events that follow (how many, when, who, to whom, and what they say) and then continues from its own output, with nothing outside it choosing the speaker. The speaker becomes an output. What the model generates must also become the world the next events come from, so a generated post must be selectable as a later actor's reply target. A system whose events only attach to real history is conditioning on the past, not simulating a population, whatever its per-event numbers say (Li & Tao, 2026; Tomašević et al., 2025; Figure 1, tested in §5.3).

A model of the whole event (§3). ACT-AWARE learns who acts and whom they answer, and writes what they say with OSim-4B (Zhou et al., 2026), a released behavioural foundation model post-trained with LoRA (Hu et al., 2022) and conditioned on the world state. The decision factors

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

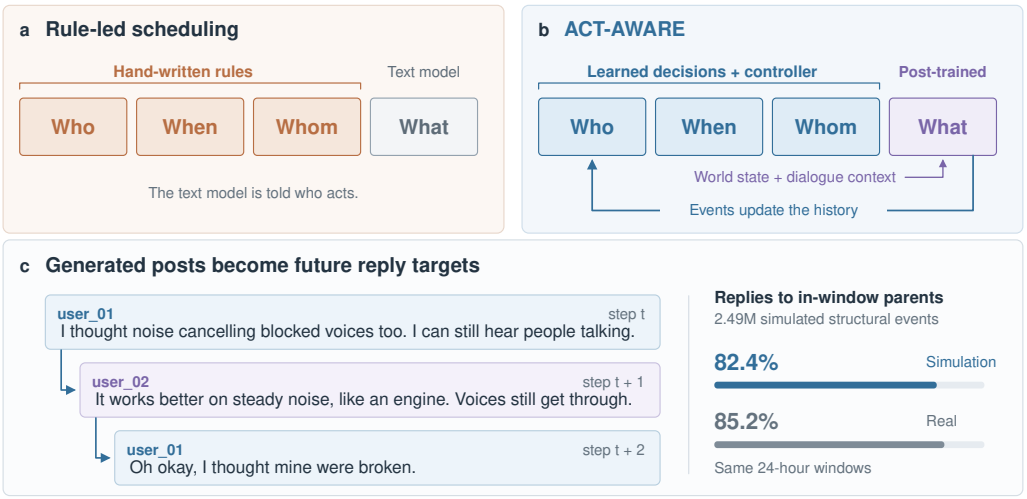

Figure 1: From rule-led scheduling to action-aware simulation. **(a)** A rule scheduler decides who acts, when, and to whom. The model supplies text. **(b)** ACT-AWARE combines learned action decisions, a rollout controller, and world-state-conditioned text generation. **(c)** Generated posts become future reply targets, illustrated with a constructed exchange.

are trained from scratch, since nothing in a speaker-conditioned checkpoint predicts who speaks. Trained jointly, the actor and reply-target factors lower held-out per-event NLL by 17.0% and 16.5% against a rule scheduler on the same events.

On who acts, the trained selector is ahead of every released model we evaluated (§4). Fifteen released models answer the same single-step problem zero-shot. Actor precision is 49.6% against 36.6% for the best of them. Within both Qwen2.5 and Llama, scores are flat with model size. Model size alone does not close the gap.

The natural rollout has a ceiling no model can lift (§5). Under a trailing-window construction the simulator draws actors from the accounts that posted within the past W. Once real events from the rollout start are hidden, every new post is written by someone already in that pool, so the set of actors the simulation can ever emit is fixed before its first step, for any W. It holds 0.0848 of the real actors at a six-hour window and 0.5114 at seven days. A wider window raises the ceiling, but the accounts it adds are far less likely to act. The controller that makes it workable is the second part of the method: it sets how many actors are drawn without reading the simulation's own output, matches how recently they posted to the measured mix, and discourages repeat selections. With this controller, the simulator reaches $2.35\times$ the six-hour ceiling from a seven-day pool over five held-out days, and its learned score beats both a lifetime-count null and a gradient-boosted forest on coverage and precision (§6).

2 The full-event simulation task

2.1 Definition

We simulate a discussion platform in 15-minute steps, short enough to preserve the burstiness that coarser steps average away (Appendix F). At step t the simulator emits a set of events E_t, each carrying an actor a_e, a minute μ_e in the step, a parent π_e (the post replied to, or none) and text x_e. With S_t the visible world state, D_t how many distinct actors emit, A_t which ones and n_u how many events actor u emits, the step factorises as

$$p(E_t \mid S_t) = \underbrace{p(D_t \mid S_t)}_{\text{how many act}} \underbrace{p(A_t \mid D_t, S_t)}_{\text{who}} \prod_{u \in A_t} \underbrace{p(n_u \mid u, S_t)}_{\text{how many each}} \times \prod_{e \in E_t} \underbrace{p(\mu_e)}_{\text{when}} \underbrace{p(\pi_e \mid a_e, S_t)}_{\text{to whom}} \underbrace{p(x_e \mid a_e, \pi_e, S_t)}_{\text{what}}, \tag{1}$$

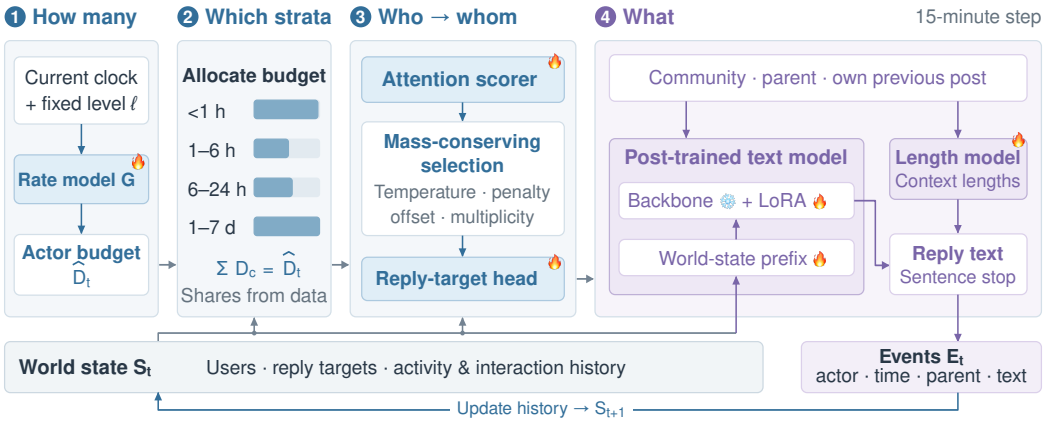

Figure 2: One 15-minute simulation step. The simulator predicts an actor budget from the current clock and a fixed pre-rollout activity level, allocates it across recency strata, selects actors and reply targets, and generates text conditioned on world state and dialogue context. Emitted events update the state for subsequent steps. Flames mark what is trained or fitted here; the snowflake marks the released backbone.

after which $S_{t+1} = S_t \cup E_t$ deterministically. Equation 1 describes how a step is drawn; its factors are fitted separately (§3). ACT-AWARE replaces the first two factors with a predicted budget $\hat{D}_t$ and a Bernoulli selection satisfying $\mathbb{E}|A_t| = \hat{D}_t$ (Appendix H). Actors and reply targets depend on the history in S_t, while x_e affects only the text of events answering it. A step can therefore be drawn structurally and its text filled in later in topological order, which makes a rollout at this scale affordable. *End to end* means all six factors are produced inside one system, four learned and two fixed, with no scheduler outside it deciding who acts or whom they answer; *closed loop* means its emissions become S_{t+1} and real events from t_{start} on are never read. The population rollouts in §§5–6 are closed-loop: 96 steps, 24 simulated hours.

2.2 Corpus

We use a public comment corpus (Baumgartner et al., 2020): 515 million events from 18.1 million authors across 50 communities, 2006 to 2023, split by time. Everything before 2019 (282 million events, 54.8%) is *enrolment*, never trained or evaluated on, so account age, lifetime count and posting history are real quantities. The remaining 233 million are training (2019-01-01 to 2021-06-30), validation (to 2021-12-31) and held-out (2022-01-01 to 2023-01-31), disjoint in that order.

Measured on the training segment over 86,879 consecutive steps, a step draws 1,579.1 events from 1,268.0 distinct actors. Of those actors 68.23% were active within the past week, so a trailing-window design can name them. A further 26.86% have been silent for over seven days (Appendix I.2), and 4.91% have never posted before. Those are per-step shares. Coverage is measured over the union across a window, a larger set with a smaller reachable fraction (§5.1).

3 The action-aware model

A behavioural foundation model such as OSim-4B (Zhou et al., 2026) is a text model. Given a speaker and a context it produces what that speaker says. Who speaks, when and which post they answer are decided elsewhere. ACT-AWARE closes that gap by emitting events from a world state, which re-enter the state as the candidates, contexts and reply targets of the events that follow (Figure 2). Six factors realise Equation 1: four learned, an estimated multiplicity table, and a fixed uniform minute. The three that condition on the world read it through a two-layer MLP encoder turning S_t into a 256-dimensional summary z.

How many, and when. A rate component G predicts the number of distinct actors emitting in $[t, t+\Delta)$ from the calendar clock and one scalar fixed before the rollout (Appendix G). A multiplicity table estimated on the training segment turns a selected actor into an event count, and the minute is drawn uniformly. **Who.** An actor head A scores every candidate from 16 features (eleven for the candidate's history, five for the step it sits in) and z_a. A second-stage hazard refines that score with the

Table 1: Per-event NLL on 5,410 held-out events, three seeds. A learned factorisation against a rule, the text term identical on both sides. The learned side is the joint-objective run; the separately fitted factors the simulator deploys are tested in closed loop instead (§6; Appendix I.6).

term	rule scheduler	learned, joint objective	gain (nats/event)	reduction
actor (who acts)	10.957	**9.090**	1.868	17.04%
reply target (to whom)	6.493	**5.420**	1.074	16.53%
both decision terms	17.451	**14.509**	2.941	16.85%

Table 2: Real-versus-generated discrimination, blind and one item at a time. AUC is the chance a judge ranks a real reply above a generated one; *matched* draws the real reply at the same length. Intervals are in Table S5.

AUC ↓	Qwen2.5-14B		Qwen2.5-32B	
where the reply ends	natural	matched	natural	matched
the factor's own stop token	0.744	0.720	0.788	0.802
length model, community only	0.660	0.601	0.690	0.654
length model, full context	**0.622**	**0.562**	**0.664**	**0.609**

candidate's recency; the deployed one is an attention scorer over per-feature tokens (Appendix I.5), and scores become a selection sized by G's total (Appendix H). **To whom.** A reply-target head B scores each (actor, parent) pair from 10 features and z_b, trained by cross-entropy over 1024 sampled parents. **What.** The text factor is OSim-4B (Qwen3) post-trained with LoRA, with z_c entering as eight learned soft-prompt tokens prepended to the token embeddings. Where a reply ends is set by a learned conditional quantile model that reads the community and the lengths of the parent and of the actor's previous post, because the factor's own stop token fires far too late (Appendix D).

Training. The three world-conditioned components are fitted in stages: the actor scorer on a training-segment holdout, the reply-target head on pools drawn by the simulator's own rule at the window it serves (Appendix E), and the text factor as a LoRA (Hu et al., 2022) over the released OSim-4B weights, then frozen while a gated world-state prefix is learned on top. A joint-objective run minimises $\lambda_a \mathcal{L}_A + \lambda_b \mathcal{L}_B + \lambda_c \mathcal{L}_C$ over all three.

Held-out gains over the rule scheduler. Table 1 compares per-event negative log-likelihood against the status quo it replaces, the same OSim-4B text factor under a rule scheduler. That scheduler is the strongest hand-specified one on the same inputs: a logistic hazard over multi-scale activity counts, best of six including a fitted self-exciting intensity, parents scored by popularity and recency (Appendix A). It runs on 5,410 held-out events, every one by an actor in neither the training nor validation segment. Per-seed intervals exclude zero on every term.

Conditioning is nearly free; optimising three losses at once is not. On 8,000 validation events the frozen checkpoint is 170.659 nats/event and a text-only LoRA 127.309. Conditioning that LoRA under one summed objective gives 153.522. In two stages, freezing the LoRA and then learning eight prefix embeddings and a projection of z behind a zero-initialised gate, it gives 128.728. Conditioning therefore costs 1.4 nats. The joint-objective run is 24.8 nats/event higher than the two-stage procedure.

Is the text it writes real? Two instruction models rate replies for human authorship, blind and one item at a time, over the 116 complete events of Appendix D and real controls. Where a reply ends accounts for much of what separates the two (Table 2). On first sentences alone the judges do not reliably separate generated text from real (AUC 0.514 to 0.529).

4 Comparison with prompted models

No released model is trained for the task of §2, so the next question is what a prompted one does with it. Fifteen take the identical single-step problem over 120 episodes from validation and are asked for the full event tuple, in the form Appendix K states. Each episode plants $n \sim \text{Uniform}\{0..4\}$ real actors, realised mean 1.90, so chance is 0.095 (the pool's own base rate is 0.0024). A prompted model is what an LLM-agent simulator could use instead of a rule to decide who acts.

Table 3: Fifteen released models and ours on the identical episodes. *actor prec* is the share of named actors that really acted, against chance 0.095; *fmt* is the parseable-answer rate (— for our arms, which return a selection). *ev/ep* is events named per episode and *count MAE* its error against the planted count; our two rows differ only in how that count is set. The oracle row is given the planted count and is shown for reference. Appendix K gives the remaining columns and the paired margins.

arm	fmt	ev/ep	count MAE	actor prec
ours, matched count	—	2.00	1.25	**0.4958**
ours, oracle count	—	1.90	0.00	0.6028
Claude Opus 4.6	0.87	1.57	1.30	0.3662
Llama-4-Maverick-17B-128E	0.93	2.81	1.56	0.2618
GPT-5	1.00	5.47	3.73	0.2320
OSim (`cmu-lti/osim-8b`)	0.61	2.12	1.90	0.2226
Qwen2.5-14B-Instruct	1.00	3.00	1.48	0.2139
Llama-3.3-70B-Instruct	1.00	5.00	3.10	0.2100
Qwen2.5-7B-Instruct	1.00	3.00	1.48	0.1972
Qwen2.5-32B-Instruct	1.00	3.08	1.52	0.1844
Qwen2.5-3B-Instruct	0.42	0.92	1.68	0.1503
Llama-3.1-405B-Instruct	1.00	7.92	6.03	0.1444
gemma-3-4b-it	0.93	5.26	3.54	0.1405
Hermes-3-Llama-3.1-8B	0.97	3.11	1.62	0.1318
Phi-4-mini-instruct	1.00	1.99	1.26	0.1292
random over the same 20	1.00	2.00	1.25	0.1125
Gemini 3.1 Pro	0.96	0.00	1.90	—
OSim-Inst (`sunweiwei/OSim-Inst-8B`)	1.00	0.00	1.90	—

On *who acts*, the trained selector is in a different regime (Table 3). At a matched count and no oracle information it scores 0.4958, 1.35× the best prompted model, Claude Opus 4.6 at 0.3662, and 4.4× the measured random arm. Margins are paired on the episodes both arms answer: over Claude +0.1146 [0.0537, 0.1763] on 104 episodes, over GPT-5 +0.2638 [0.2117, 0.3159] on 120, and over OSim-8B +0.2842 [0.2066, 0.3644] on 73.

Larger models do not close the gap. Qwen2.5 at 3B, 7B, 14B and 32B scores 0.150, 0.197, 0.214, 0.184; Llama at 70B and 405B, 0.210 and 0.144. Scores are flat across the 135× parameter range the two families span. Nor does behavioural post-training supply it: OSim-8B reaches 0.2226. What none of them has is a component that decides who speaks, here under a million parameters. Ziems et al. (2026) find most social-simulation abilities do scale with compute, and isolate the behavioural sub-tasks that stay flat from 0.5B to 8B even when fine-tuned.

5 Closed-loop rollout

Driving a single-event model as a population means deciding, at every step, how much to emit, how to spread it over a pool 8.8× larger than a six-hour window gives, and how to stop the distribution drifting once the model reads its own output (Figure 2). At this scale, these inference-time choices determine the reachable actor set, emitted actor-recency composition, and feedback dynamics.

5.1 Population closure

Proposition 1 (Population closure)**.** *Consider a rollout that observes real events strictly before t_{start} and only its own generated events thereafter. Let $P_t :=$ unique(authors of visible events in $[t-W, t)$) be the candidate actor pool at step t, and assume every event generated at step s is authored by an actor selected from P_s. Let $R_W :=$ unique(authors in $[t_{\text{start}} - W, t_{\text{start}})$), let A_{emit} be the actors emitted during the rollout, and let A_{real} be the distinct actors of the corresponding real horizon. Then $P_t \subseteq R_W$ for every $t \geq t_{\text{start}}$, and distinct-actor coverage obeys $|A_{\text{emit}} \cap A_{\text{real}}|/|A_{\text{real}}| \leq |R_W \cap A_{\text{real}}|/|A_{\text{real}}|$, regardless of model, decoder and training objective.*

Proof. At t_{start}, $P_{t_{\text{start}}} = R_W$. Suppose $P_u \subseteq R_W$ for every $u \leq s$. Pre-rollout real events can only leave the trailing window, while every event generated at a step $u \leq s$ is authored by an actor in P_u,

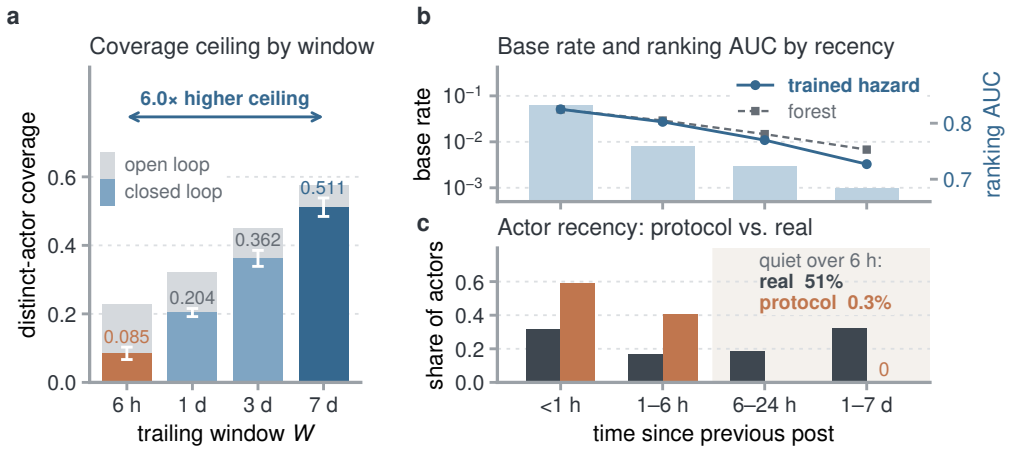

Figure 3: Why population rollout is hard before any text is generated. **(a)** Proposition 1's closed-loop ceiling at four trailing-window widths, mean ± s.d. over the five held-out windows; pale extensions show the corresponding open-loop ceiling, whose pool refreshes from real activity. **(b)** base rate (bars) and ranking AUC (lines) across the four recency strata, computed on 1.37×10^7 labelled candidate–step rows from the training segment; solid and dashed lines denote the trained hazard and a per-stratum gradient-boosted forest using the same features. **(c)** held-out actor-recency composition under the 6 h protocol, compared with the real composition measured on the training segment.

so the next pool is again a subset of R_W; the claim follows by induction. Hence $A_{\text{emit}} \subseteq R_W$, which gives the stated bound. □

The ceiling depends on W. Across the five held-out windows the mean ceilings are 0.0848, 0.2039, 0.3620 and 0.5114 for $W = 6$ h, 1 d, 3 d and 7 d, a 6.0× range, and at every width the corresponding open-loop ceiling is higher (Figure 3a, Table S2). ACT-AWARE remains subject to Proposition 1 and obtains a higher ceiling by using a seven-day pool; the remaining subsections address the resulting selection and feedback problems. At $W = 6$ h coverage plateaus below its own ceiling by step 24 (Figure 5a), showing that selection dynamics bind before candidate availability.

5.2 Consequences of the wide pool

Aggregate precision measures composition as much as ranking. Per-selection precision is a weighted average over strata differing 64× in base rate and only 1.13× in how well they can be ranked (Figure 3b), the weights set by which strata a protocol emits into (Figure 3c). Our allocation therefore pins the emitted composition to the measured one, and every precision comparison below either holds composition fixed across arms or is reported within strata (Table 5; Appendix B).

The hazard has to be trained on the distribution it is served. It is fitted on features from real histories, while by step 96 every short-window feature it reads is self-generated. One round of dataset aggregation (DAgger; Ross et al., 2011) on eight training-segment windows closes that gap (Appendix C). Recursive closure depends on applying the same principle to the reply-target head.

5.3 Recursive closure

Emitting the right amount of the right activity does not make the simulator a population. Its events have to become the world the next ones come from. Of the replies real users wrote in the five held-out windows (§6.1), 85.18% named a parent created in that same window; the simulator reproduces that rate at 82.35% (Figure 4). The control that makes this non-trivial uses the same architecture, fitted on 6 h pools and served on 7 d pools. It never attaches to a generated post, a support failure rather than a preference against generated posts (Figure S3). Matching the head's training and serving pools yields this attachment rate, with changes of −0.0046 in coverage and −0.0028 in precision relative to the control (Table S6).

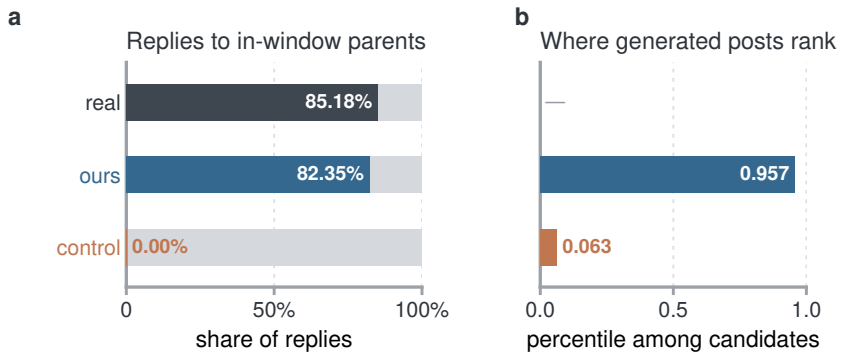

Figure 4: Recursive closure. **(a)** share of replies to a parent created inside the window; **(b)** the percentile at which a reply-target head ranks generated candidates. Control: the same head fitted on 6 h pools.

The run has two tiers. Structurally it runs at population scale: 2,494,058 events in 440,145 reply trees, 2,053,913 answering a post the same run generated. Text is then written for 413 of those trees *in full* (1,575 events), drawn uniformly at random, each from its root down. Of those, 1,162 answer a generated post this run also wrote, in chains reaching depth 11. At depth 4 and beyond a reply follows at least three generated events up its own chain, and shows no detectable loss of variety against depth 1: distinct-3 of 0.9824 against 0.9819, in a comparison with 80% power to detect a 1.76% change. No sign appears of the convergence *between* replies that recursive generation is known for (Appendix E).

6 Population-level fidelity

6.1 Setup

Protocol and arms were fixed before the five primary held-out windows were run. The four model arms, **T** (ours), **U** (no selection controls) and the nulls $\mathbf{N_r}$ and $\mathbf{N_u}$, share the corpus, windows, seeds, frozen encoders and heads, multiplicity table, attachment decoder, reply-target head, and, critically, the same $\hat{D}_t$ sequence, computed once per window. **V** replays the trailing-window construction with its own budget, multiplicity, attachment rule and reply-target head, but it reads our actor scorer, so the contrast holds the ranking model fixed and changes only the protocol. The nulls receive **T**'s temperature, repeat-selection penalty and composition constraint, so they isolate the score and nothing else. Table S14 defines each arm; Appendix I reports validation evidence for the design choices.

Coverage is distinct emitted actors that really acted, over all real actors in the window; *precision*, per-step hits over per-step selections; *volume*, emitted over real events; *posts per actor*, emitted events over distinct emitted actors against a real 1.91; *composition*, the largest relative deviation of the four emitted recency shares from the measured ones. Three seeds on five windows give fifteen runs per arm. Contrasts are paired by (seed, window) in a 10^4-resample bootstrap; Table S15 also reports window-level inference over the five replication units.

6.2 Coverage and precision

Over five 24-hour held-out windows the model reaches coverage 0.1996 at per-selection precision 0.0345 (Table 4, Figure 5). A trailing 6 h pool has a ceiling of 0.0848 here whatever model drives it. On the seven-day pool the model reaches 2.35× that ceiling, where a random ranking reaches 1.27× and lifetime post count 1.85×. The 6 h protocol reaches 0.0469, 55% of its own ceiling. The run holds the maximum relative deviation of the emitted recency shares to 0.0065 and reaches precision 0.0793 in the freshest stratum, where the trailing-window protocol places most of its emission (Table 5).

6.3 Score ablations

The contrast that isolates the learned actor score is against $\mathbf{N_r}$: **T** with the trained head replaced by lifetime post count, everything else held fixed. On it **T** is higher on both axes, +0.0428 coverage [+0.0401, +0.0455] and +0.0141 precision [+0.0129, +0.0151], and higher in all four recency strata (Ta-

Table 4: All arms share five held-out 24-hour windows and three seeds, matched by (seed, window). **U** turns off the three selection controls (temperature, repeat-selection penalty and composition constraint), the nulls replace the actor score, and **V** is the trailing-window protocol, driven by our scorer. Relative to **U**, the controls change coverage by $+0.0215$ and precision by $+0.0170$. **T** leads every arm on both axes in each of the five windows (Table S16); paired contrasts are in Table S15.

	arm	coverage ↑	precision ↑	posts/actor	vol.
T	**ours**	**0.1996**	**0.0345**	2.89	0.994
U	no selection controls	0.1781	0.0174	2.34	0.950
$\mathbf{N_r}$	null: rate	0.1568	0.0204	3.48	1.100
$\mathbf{N_u}$	null: uniform	0.1079	0.0042	2.41	0.913
V	6 h protocol	0.0469	0.0309	14.14	0.776
	real	1.0	—	1.91	1.000



Figure 5: The held-out result, on the five windows of §6.1. **(a)** cumulative distinct-actor coverage over a 24 h closed-loop rollout, averaged over 15 (seed, window) pairs; the 6 h closed-loop ceiling is the only model-independent quantity in the panel. **(b)** the coverage/precision plane: the five arms of Table 4 (bars: 95% interval over the 15 pairs) against a validation sweep of the temperature alone.

ble 5), so the gain is in ranking, with the emitted composition matched. Against the uniform null it is $+0.0917$ and $+0.0303$.

Against a learned competitor. The competitor is a per-stratum gradient-boosted forest trained on the same features and the same data as our scorer (§5.2). Matching the data matters: on validation the aggregated rows are worth $+0.0304$ coverage to the forest, about four times its margin over the MLP head (Appendix I.4). On six held-out windows disjoint from the five above, the deployed scorer clears it on both axes by $+0.0103$ coverage [+0.0068, +0.0135] and $+0.0034$ precision [+0.0022, +0.0046], with intervals clustered by window (Table 6).

On validation, using the predicted total instead of the *real* per-step actor count changes neither coverage nor precision measurably ($+0.0004$ and $+0.0001$, both intervals spanning zero). No arm diverged; **T**'s worst single step emitted $1.06\times$ real volume (Appendix J.1).

6.4 Protocol ablation: the trailing-window construction

Arm **V** reads the *real* per-step actor count as a budget **T** never sees. **T** reaches $4.26\times$ **V**'s distinct-actor coverage, a difference of $+0.1527$ with a five-window paired-t interval of [+0.1290, +0.1764]. The precision difference is $+0.0035$ and positive in every window, but its five-window paired-t interval includes zero (Table S15). Within each stratum **V** selects from, **T**'s precision is higher (Table 5). Most of the coverage gap is imposed by **V**'s six-hour pool. Proposition 1 confines each **V** rollout to its initial six-hour actor set, averaging 10,035 users across the five windows. Over 96 steps, **V** emits 14.14 posts per actor, versus 1.91 in the real data.

7 Related work

LLM-agent social simulation. Generative-agent systems and their descendants (Park et al., 2023; Yang et al., 2024; Piao et al., 2025; Tang et al., 2025; Wang et al., 2025; Zhang et al., 2025; Ng &

Table 5: Precision within each recency stratum on the five held-out windows, and the largest relative deviation (*max rel. dev.*) of the emitted recency shares from the measured composition. —: no one selected.

	max rel.	precision within stratum			
	dev.	<1 h	1–6 h	6–24 h	1–7 d
T	0.0065	**0.0793**	**0.0289**	**0.0143**	**0.0054**
U	0.6517	0.0662	0.0202	0.0105	0.0035
$\mathbf{N_r}$	0.0062	0.0449	0.0170	0.0104	0.0042
$\mathbf{N_u}$	0.0072	0.0056	0.0052	0.0050	0.0018
V	1.4247	0.0411	0.0159	0.0048	—

Table 6: The actor scorer substituted three ways inside the same controller, on six held-out windows disjoint from the five of §6.1; every other component is **T**'s. Intervals, volume and per-actor counts in Table S13.

actor scorer	coverage	precision
MLP scorer	0.1885	0.0313
per-stratum forest, same data	0.1944	0.0334
attention scorer (ours)	**0.2047**	**0.0367**

Carley, 2026) give each agent a language model and memory and route turns from outside it (see Cau et al., 2026, for a survey). We make that routing part of the model. Zhang et al. (2026) calibrate that outside rule to recover bursty timing, and Gershman (2026) adds more scaffolding instead. Buitrago López et al. (2026) and Guo et al. (2026) ask how far such systems reach at one person's scale; Proposition 1 bounds how far a whole class of protocols reaches at population scale. Zhou et al. (2026), Shi et al. (2025) and Gabrielsson (2026) train or scale behavioural models, but a speaker or user is an input to all of them, and Qiu et al. (2025) predict which action a *given* user takes, so who responds at all is supplied by the data. That is the factor this paper trains.

Who acts, at two scales. A long line of work predicts the next speaker in a bounded conversation (Nonomura & Mori, 2024; Gatti de Bayser et al., 2019; Bhagtani et al., 2026; Patamia et al., 2026; Navarro et al., 2025) with a roster of a few participants, while Fu et al. (2025) and Münker et al. (2026) predict the next *item* without modelling which user produces it. At population scale the roster is 10^7 accounts, the base rate 10^{-3}, and a selector's accuracy is dominated by which strata the protocol asks it about.

Cascade and activity models. Point processes (Hawkes, 1971; Perra et al., 2012), learned cascade and discussion models (Horawalavithana et al., 2020; Ling et al., 2020; Zayats & Ostendorf, 2018) and measurements of diffusion structure (Goel et al., 2016) all concern intensity or growth; Proposition 1 instead constrains the mark distribution. On validation, a per-user self-exciting intensity picks who acts no better than the best activity rule, and about half of the actor head's gain over that rule comes from replies a candidate has just received and how busy its communities are, which such an intensity does not read (Appendix A). Closest is Deng et al. (2026), where a multivariate Hawkes process picks which node acts and which earlier outputs enter its prompt. Its coupling is pairwise, so the node set is bounded and it runs on news cascades, while the factor here ranks candidates from 18.1M accounts and picks an explicit reply parent from a live pool. Recursively generated data is known to degrade a model (Alemohammad et al., 2024; Shumailov et al., 2024). That literature concerns weights trained on generations. Here the *context* is generated and the weights fixed. Even with fixed weights, a per-user hazard that reads the generated context reaches $16.4\times$ the real volume within six hours, which is why the controller sets the total without reading it (Appendix H).

8 Conclusion

Told who is speaking, a behavioural foundation model does only part of the job; full-event simulation asks for the rest. ACT-AWARE models the whole event: the two factors deciding who acts and whom they answer are learned rather than ruled, and a data-calibrated controller serves them as a population from a 7 d pool at $2.35\times$ the closed-loop coverage ceiling of a 6 h pool. Structurally, its events become the world the next ones come from, with 82.4% of its replies landing on a parent it generated itself. A coverage number is only readable against the ceiling its pool implies.

Reproducibility

The corpus is public (Baumgartner et al., 2020). Appendix L gives every architecture, feature list, optimiser setting and training schedule for all six factors, and the prompt and schema the released models were given; Appendix M gives the six stages in the order they run. The training, validation and held-out splits are disjoint in time, and the five windows carrying the population-level results and the six carrying the scorer comparison are disjoint from each other and had not been rolled out before the runs reported here. Appendix J.2 reports each of the five primary windows separately.

Ethics

The model was trained on a public corpus of comments (Baumgartner et al., 2020). Because it learns from individual posting histories, results are reported at the population level, and the text it generates was used only for evaluation and never posted. Every example shown is model output or constructed, handles are pseudonymised, and no user data is released.

AI use statement

We used generative AI tools for editing the paper for readability, drafting parts of it, searching for and summarising related literature, writing and running analysis code, and interpreting results. We have reviewed all AI-assisted work, and we take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

References

Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard G. Baraniuk. Self-consuming generative models go MAD. In *International Conference on Learning Representations*, 2024.

Robert L. Axtell. Effects of interaction topology and activation regime in several multi-agent systems. In Scott Moss and Paul Davidsson (eds.), *Multi-Agent-Based Simulation*, volume 1979 of *Lecture Notes in Computer Science*, pp. 33–48. Springer, 2000. doi: 10.1007/3-540-44561-7_3.

Albert-László Barabási. The origin of bursts and heavy tails in human dynamics. *Nature*, 435: 207–211, 2005. doi: 10.1038/nature03459.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. The Pushshift Reddit dataset. In *Proceedings of the International AAAI Conference on Web and Social Media (ICWSM)*, volume 14, pp. 830–839, 2020. arXiv:2001.08435.

Kratika Bhagtani, Mrinal Anand, Yu Chen Xu, and Amit Kumar Singh Yadav. Speak or stay silent: Context-aware turn-taking in multi-party dialogue. *arXiv preprint arXiv:2603.11409*, 2026.

Alejandro Buitrago López, Javier Pastor-Galindo, and José A. Ruipérez-Valiente. Should LLM agents decide in social simulations? Comparing finite-state and LLM-based decision policies. *arXiv preprint arXiv:2606.12369*, 2026.

Erica Cau, Andrea Failla, Valentina Pansanella, and Giulio Rossetti. Social simulations: From agent-based modeling to digital twins. *arXiv preprint arXiv:2607.13693*, 2026. Entry for the Encyclopedia of Social Network Analysis and Mining.

Zewei Deng, Tinghan Ye, and Liyan Xie. HawkesLLM: Semantic uncertainty propagation in agentic text simulation. In *ICML 2026 Workshop on Statistical Frameworks for Uncertainty in Agentic Systems*, 2026. arXiv:2605.23043.

Bokang Fu, Jiahao Wang, Xiaojing Liu, and Yuli Liu. Can we predict the next question? A collaborative filtering approach to modeling user behavior. *arXiv preprint arXiv:2511.12949*, 2025.

Rickard Brüel Gabrielsson. Scaling laws for behavioral foundation models over user event sequences. *arXiv preprint arXiv:2606.05257*, 2026.

Maira Gatti de Bayser, Paulo Cavalin, Claudio Pinhanez, and Bianca Zadrozny. Learning multi-party turn-taking models from dialogue logs. *arXiv preprint arXiv:1907.02090*, 2019.

Anatole Gershman. Classic AI scaffolding for LLM social agents. *arXiv preprint arXiv:2609.01167*, 2026.

Sharad Goel, Ashton Anderson, Jake Hofman, and Duncan J. Watts. The structural virality of online diffusion. *Management Science*, 62(1):180–196, 2016. doi: 10.1287/mnsc.2015.2158.

Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Yury Gorishniy, Ivan Rubachev, and Artem Babenko. On embeddings for numerical features in tabular deep learning. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Minghao Guo, Ziyi Ye, Wujiang Xu, Xi Zhu, Wenyue Hua, and Dimitris N. Metaxas. Individual Turing test: A case study of LLM-based simulation using longitudinal personal data. *arXiv preprint arXiv:2603.01289*, 2026.

Alan G. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971. doi: 10.1093/biomet/58.1.83.

Sameera Horawalavithana, John Skvoretz, and Adriana Iamnitchi. Cascade-LSTM: Predicting information cascades using deep neural networks. *arXiv preprint arXiv:2004.12373*, 2020.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Yiming Li and Dacheng Tao. AI agents alone are not (yet) sufficient for social simulation. *arXiv preprint arXiv:2603.00113*, 2026.

Chen Ling, Ruiqi Wang, and Guangmo Tong. SocialGrid: A TCN-enhanced method for online discussion forecasting. *arXiv preprint arXiv:2003.07189*, 2020.

Fred Mubang and Lawrence O. Hall. Simulating user-level Twitter activity with XGBoost and probabilistic hybrid models. *arXiv preprint arXiv:2202.08964*, 2022.

Simon Münker, Nils Schwager, Kai Kugler, Michael Heseltine, and Achim Rettinger. Next reply prediction X dataset: Linguistic discrepancies in naively generated content. *arXiv preprint arXiv:2602.19177*, 2026.

Madeline Navarro, Lisa O'Bryan, and Santiago Segarra. Learning time-varying turn-taking behavior in group conversations. *arXiv preprint arXiv:2510.18649*, 2025.

Lynnette Hui Xian Ng and Kathleen M. Carley. Modeling social dynamics with an LLM-enabled agent based network-dynamic (LAND) model. *arXiv preprint arXiv:2608.00929*, 2026.

Ryota Nonomura and Hiroki Mori. Who speaks next? Multi-party AI discussion leveraging the systematics of turn-taking in murder mystery games. *arXiv preprint arXiv:2412.04937*, 2024.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. doi: 10.1145/3586183.3606763.

Rutherford A. Patamia, Ming Liu, Wei Luo, Favour Ekong, and Akan Cosgun. Fast when, careful who: Dual-process multiparty turn-taking with diffusion augmentation. *arXiv preprint arXiv:2606.16568*, 2026.

Nicola Perra, Bruno Gonçalves, Romualdo Pastor-Satorras, and Alessandro Vespignani. Activity driven modeling of time varying networks. *Scientific Reports*, 2:469, 2012. doi: 10.1038/srep00469.

Jinghua Piao, Yuwei Yan, Jun Zhang, Nian Li, Junbo Yan, Xiaochong Lan, Zhihong Lu, Zhiheng Zheng, Jing Yi Wang, Di Zhou, Chen Gao, Fengli Xu, Fang Zhang, Ke Rong, Jun Su, and Yong Li. AgentSociety: Large-scale simulation of LLM-driven generative agents advances understanding of human behaviors and society. *arXiv preprint arXiv:2502.08691*, 2025.

Zhongyi Qiu, Hanjia Lyu, Wei Xiong, and Jiebo Luo. Can LLMs simulate social media engagement? A study on action-guided response generation. *arXiv preprint arXiv:2502.12073*, 2025.

Stéphane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

Quan Shi, Carlos E. Jimenez, Stephen Dong, Brian Seo, Caden Yao, Adam Kelch, and Karthik Narasimhan. IMPersona: Evaluating individual level LM impersonation. *arXiv preprint arXiv:2504.04332*, 2025.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759, 2024. doi: 10.1038/s41586-024-07566-y.

Jiakai Tang, Heyang Gao, Xuchen Pan, Lei Wang, Haoran Tan, Dawei Gao, Yushuo Chen, Xu Chen, Yankai Lin, Yaliang Li, Bolin Ding, Jingren Zhou, Jun Wang, and Ji-Rong Wen. GenSim: A general social simulation platform with large language model based agents. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pp. 143–150, 2025. doi: 10.18653/v1/2025.naacl-demo.15.

Aleksandar Tomašević, Darja Cvetković, Sara Major, Slobodan Maletić, Miroslav Anđelković, Ana Vranić, Boris Stupovski, Dušan Vudragović, Aleksandar Bogojević, and Marija Mitrović Dankulov. Towards operational validation of LLM-agent social simulations: A replicated study of a Reddit-like technology forum. *arXiv preprint arXiv:2508.21740*, 2025.

Lei Wang, Heyang Gao, Xiaohe Bo, Xu Chen, and Ji-Rong Wen. YuLan-OneSim: Towards the next generation of social simulator with large language models. *arXiv preprint arXiv:2505.07581*, 2025.

Ziyi Yang, Zaibin Zhang, Zirui Zheng, Yuxian Jiang, Ziyue Gan, Zhiyu Wang, Zijian Ling, Jinsong Chen, Martz Ma, Bowen Dong, Prateek Gupta, Shuyue Hu, Zhenfei Yin, Guohao Li, Xu Jia, Lijun Wang, Bernard Ghanem, Huchuan Lu, Chaochao Lu, Wanli Ouyang, Yu Qiao, Philip Torr, and Jing Shao. OASIS: Open agent social interaction simulations with one million agents. *arXiv preprint arXiv:2411.11581*, 2024.

Victoria Zayats and Mari Ostendorf. Conversation modeling on Reddit using a graph-structured LSTM. *Transactions of the Association for Computational Linguistics*, 6:121–132, 2018. doi: 10.1162/tacl_a_00009.

Anping Zhang, Yang Tan, Yuanbo Tang, Huaze Tang, Qiuhua Ye, Marta C. Gonzalez, and Yang Li. Toward temporal realism in city-scale crisis response simulation using LLM agents. *arXiv preprint arXiv:2606.19904*, 2026.

Xinnong Zhang, Jiayu Lin, Xinyi Mou, Shiyue Yang, Xiawei Liu, Libo Sun, Hanjia Lyu, Yihang Yang, Weihong Qi, Yue Chen, et al. SocioVerse: A world model for social simulation powered by LLM agents and a pool of 10 million real-world users. *arXiv preprint arXiv:2504.10157*, 2025.

Xuhui Zhou, Weiwei Sun, Weihua Du, Jiarui Liu, Haojia Sun, Qianou Ma, Tongshuang Wu, Yiming Yang, and Maarten Sap. OdysSim: Building foundation models for human behavior simulation. *arXiv preprint arXiv:2606.14199*, 2026.

Caleb Ziems, William Held, Su Doga Karaca, David Grusky, Tatsunori Hashimoto, and Diyi Yang. Will scaling improve social simulation with LLMs? *arXiv preprint arXiv:2607.02464*, 2026.

Supplementary Material

Contents

A The held-out NLL comparison **13**

B Aggregate precision **14**
B.1 Precision within matched strata 15

C Dataset aggregation for the hazard **15**

D Text generation and stopping **15**

E Recursive closure **18**

F Burstiness and recency composition **22**

G The global rate model **22**

H Composition allocation **23**

I Design decisions **23**
I.1 Resampling and temperature 23
I.2 The dormant-user channel 24
I.3 Composition pinning 24
I.4 Training distribution and scorer class 24
I.5 The attention scorer 25
I.6 The scorer substitution 26

J Arms, paired contrasts and controls **27**
J.1 Additional controls 27
J.2 Per-window results 28

K Prompted models on the same task **28**
K.1 Setup and scope of the comparison 28
K.2 Results 29

L Architectures, features and training settings **30**

M Reproduction **32**

A The held-out NLL comparison

§3 compares per-event negative log-likelihood against the status quo the model replaces. This section gives the baseline's construction and the event filter; Figure S1 plots both sides of the comparison.

How the rule baseline was built. Its actor side is a logistic hazard over multi-scale activity counts (at 1 h, 6 h, 24 h and 7 d, plus enrolment and recency) selected on the training segment as the best of six candidates and then frozen, softmaxed over the same candidate pool the model reads. The field it was selected from includes a fitted exponential-kernel self-exciting intensity, which it beats on validation mean average precision, 0.2468 against 0.2390. Parents are scored by popularity and recency, $\log(1 + \text{replies}) - \log(1 + \text{age})$, over the same 1024 candidates; substituting a *fitted* parent scorer moves the reply-target term by 0.042 nats on validation. So the rule is the strongest hand-specified scheduler on the same inputs.

What the learned head reads that the rule does not. On validation we rank the full candidate pool at each step and ask what share of the step's actors never seen acting in training land in its top B_t, B_t being the step's real actor count. That share is 0.2941 for the per-user self-exciting intensity, 0.2990 for the rule, 0.3170 for a learned head given the activity features alone, and 0.3338 with all of them. The learned head is the development version of the actor head. It reads the same features as the deployed one (Appendix L), three of which the rule does not: `inbound_recency` and `inbound_6h`,

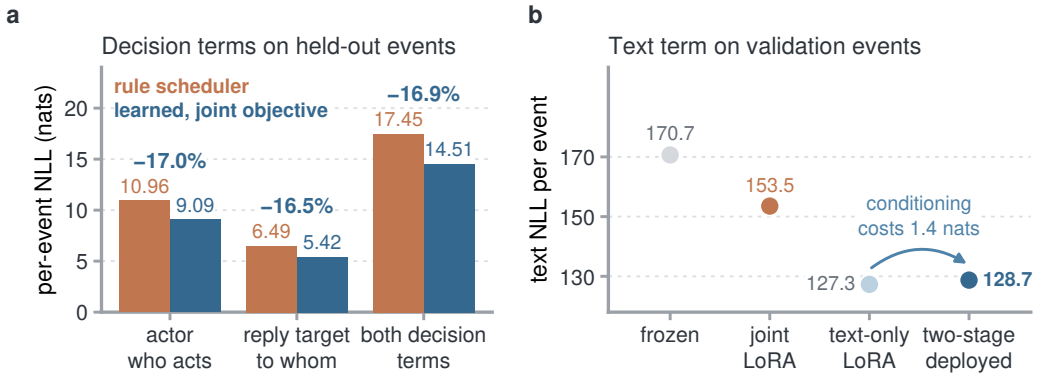

Figure S1: (a) the two decision terms, per event, on 5,410 held-out events. **(b)** the text factor on validation, one assembly per arm on the same 8,000 events. The deployed two-stage factor reaches 128.728 against the joint run's 153.522, so conditioning costs 1.4 nats rather than the 26.2 that run suggests.

Table S1: Strata of the seven-day candidate pool, measured on the training segment. Base rate spans 64×; ranking AUC spans 1.13×. A gradient-boosted forest on the same features improves AUC by at most 0.026. *lift @ top 1%* is the hazard's precision in the top 1% of a stratum's rows over that stratum's base rate.

time since previous post	candidate rows	base rate	AUC (hazard)	AUC (forest)	lift @ top 1%
<1 h	683,293	0.0618	0.8249	0.8248	15.5
1–6 h	878,340	0.0081	0.8025	0.8048	16.0
6–24 h	2,109,337	0.0030	0.7699	0.7806	11.7
1–7 d	10,029,767	0.0010	0.7269	0.7529	8.1

how recently and how often the candidate has just been replied to, and `community_affinity`, how busy its communities are right now. Those three add +0.0167 [+0.0154, +0.0180] of its +0.0348 gain over the rule. The gain holds in every activity quartile, from +0.035 to +0.046, and is positive in all six communities with enough rows to score, four with intervals excluding zero.

Which events are scored, and why there are 5,410 of them. Of 60,000 candidate step-events, 12,706 (21.18%) have an actor who never posted in the training or the validation segment. Of those, the actor must appear in its own step's candidate pool, which holds for 90.24%, and the event's own text must be at least ten characters. What survives is 5,410 events, 42.58% of the held-out-person candidates. The reply-target block of the text prompt is populated only where the parent is itself a comment.

Why the text term is held common. Text NLL is a per-event total over 49.15 target tokens on average (139.525 nats against 5.4 to 11.0 for a decision term, 13 to 26 times larger), so it would swamp any statement about their sum. The common text term is identical in both columns, so no text difference enters the decision comparison.

B Aggregate precision

Coverage has a protocol ceiling; the natural complement is per-selection precision, $\sum_t |\text{emitted}_t \cap \text{truth}_t| / \sum_t |\text{emitted}_t|$. It has a subtler problem.

Stratify the candidate pool by time since the candidate's previous post. On 1.37×10^7 labelled candidate–step rows from the training segment, the four strata differ enormously in difficulty and hardly at all in how well they can be ranked (Table S1, and Figure 3b):

The consequence is a methodological one. Aggregate precision is $\sum_c w_c \cdot \text{prec}_c$ with w_c the share of emission mass a protocol places in stratum c. Because prec_c spans an order of magnitude across strata (0.0793 to 0.0054 for the treatment) while achievable ranking quality spans 1.13×, **a difference in aggregate precision between two protocols is mostly a difference in** w. Comparing them as though it were a difference in ranking is a category error.

Figure 3c compares the emitted composition of the 6 h protocol with the real one. Reality places 31.7% of its actors in the freshest stratum; the trailing-window protocol places 59.3% there and essentially none in the two oldest strata. Its precision is an average over a population that is three-fifths burst users.

Two corollaries shape the rest of the paper. *For evaluation*: precision must be compared either against nulls matched on composition, or within a matched stratum. Both appear in §6. *For design*: if composition determines aggregate precision, it should be a modelled quantity rather than a by-product, which is what §H does.

Table S2: The closed-loop ceiling of Proposition 1 at four window widths, on the same five held-out windows as §6 (± s.d. across windows). The open-loop column lets the pool refresh with real activity, and shows what closing the loop costs at each width.

window W	closed-loop ceiling	open loop, for comparison
6 h	0.0848 ± 0.0181	0.2286
1 d	0.2039 ± 0.0118	0.3211
3 d	0.3620 ± 0.0233	0.4518
7 d	$\mathbf{0.5114} \pm 0.0270$	0.5760

B.1 Precision within matched strata

Table S3 checks this argument on the held-out data. The trailing-window protocol puts 59.3% of its actors in the stratum with the highest base rate; our allocation holds the measured composition to 0.65% across 96 closed-loop steps, where **U**, which leaves the composition free, drifts to 65.2% and the trailing-window protocol itself to 142.5% maximum relative deviation. Within *every* matched stratum our precision is higher, by $1.93\times$ in the burst stratum where the trailing-window protocol spends most of its mass, which is the comparison the argument above says to make.

C Dataset aggregation for the hazard

The hazard head H of Appendix L is trained on the training segment. Trained that way it only ever sees features computed from *real* histories, while at step 96 of a rollout every short-window feature it reads is entirely self-generated.

We close that gap with one round of dataset aggregation. We run the simulator on eight training-segment windows, harvest at each step the features it actually computes on its own drifted world, label them with what really happened in the training segment, and refit H on the union of the real-history episodes and the harvested ones (27.5% harvested). Nothing else changes: same architecture, same loss, same importance weights. The measured effect on validation is $+0.0063$ coverage [+0.0054, +0.0071] and $+0.0017$ precision [+0.0014, +0.0020] over the identical configuration with the original head. A second round, harvested under the round-one policy and run at full scale, gives $+0.0011$ coverage [+0.0004, +0.0017] and -0.0004 precision [–0.0006, –0.0001] relative to round one: a wash, so the treatment uses exactly one.

The final simulator uses the attention scorer evaluated in §I.6, fitted on the same aggregated rows (Appendix L).

D Text generation and stopping

The population-level metrics in §6 are structural (event volume and composition, who acts, when, and whom they answer) and come from rollouts that do not decode text. Here the full chain runs once, on validation, with every factor in place, used as trained and with nothing tuned for this run: two windows, eight steps each, seeded only with real events strictly before the window start.

Two tiers, reported separately because they cover different populations. The **structural** tier is the whole population: G sizes each step, allocation splits the total across the four recency strata, A and H with the per-stratum bisection choose the actors, the multiplicity table gives each of them an event count, and B picks each event's reply target. It produces 28,781 events carrying an actor, a

Table S3: Emitted composition and per-stratum precision on the held-out segment. The target m is the composition measured on the training segment (§F) and is what the allocator is pinned to. The nulls are composition-pinned too, so the composition contrast isolates the protocol and the constraint.

	share of emitted actors				max rel.	precision within stratum			
	<1 h	1–6 h	6–24 h	1–7 d	dev.	<1 h	1–6 h	6–24 h	1–7 d
target m	0.3165	0.1701	0.1888	0.3246	—	—	—	—	—
T	0.3165	0.1697	0.1888	0.3250	0.0065	**0.0793**	**0.0289**	**0.0143**	**0.0054**
U	0.1359	0.2587	0.2228	0.3826	0.6517	0.0662	0.0202	0.0105	0.0035
$\mathbf{N_r}$	0.3165	0.1706	0.1884	0.3245	0.0062	0.0449	0.0170	0.0104	0.0042
$\mathbf{N_u}$	0.3169	0.1701	0.1889	0.3242	0.0072	0.0056	0.0052	0.0050	0.0018
V	0.5928	0.4047	0.0025	0.0000	1.4247	0.0411	0.0159	0.0048	—

minute and a reply target, and no text. The **complete** tier takes a cohort of 28 of those actors (14 per window, drawn among those the simulator selected at least three times, so that the text loop is actually exercised) and generates every one of their events in full, in chronological order, up to six each. That is 116 complete events, each written by the LoRA-adapted OSim-4B text factor prompted with the community name, the selected parent's text, the step context as eight soft-prompt tokens, and the actor's own previous post. For 88 of the 116 that previous post is text *this run generated*, so the text factor is reading its own output rather than real history; the other 28 are each actor's first event, where no generated history exists yet.

Causality. Every read of real history is capped at the window start, as both candidate-pool builders do, at $\min(t, t_{\text{start}})$, so no prompt can contain a post published after the rollout began. So that the text stage runs on exactly the world the simulator produced, it replays the emitted stream and recomputes each step's context vector event for event.

What the base checkpoint has seen. The text factor adapts a released model, so the question of overlap with our corpus has a specific answer worth stating. OSim-4B's midtraining corpus includes a component drawn from the same fifty communities we use, over 2006–2023, which covers our evaluation period. It is drawn from the *submissions* of those communities; our corpus, the events the model is trained on, and every reply it generates or a judge scores here are *comments*. The two are disjoint sets of objects from the same public archive. What can overlap is context rather than target: where a generated reply answers a root post, that post's text may have been seen. The decision factors are unaffected (they are fitted from scratch on features) and *held out* throughout this paper means held out from the splits of §2.2.

What the text factor is worth. All three blocks of Table S5 score the two-stage factor the simulator deploys, on the same 116 events.

The factor knows where a reply *can* end. Scored on 2,000 held-out replies, its end-of-sequence token sits at median rank 4 of 151,936 at a true reply ending and at median rank 151,808 inside one: the signal is close to exact. What it does not do is commit: at median probability 0.051 it fires about one time in twenty, so left alone it runs past the ending it has identified. That gap between knowing and committing is what the length factor closes.

The factor emits a stop token where it means the reply to end (114 of 116 recorded replies end at one, and where it fires predicts the reply's length at Spearman 0.79), and each reply is cut at its first one. It stops late, a mean of 542 characters against a real 202. We report the whole reply, the unit the system actually emits, and the first sentence, which separates the writing from where it ends.

Each generated reply is paired with a real reply drawn from the same community, the same 24 h window, and a parent that also predates the window start, so topic, period and parent age are matched; a second real sample is additionally matched on length. Two instruction models, Qwen2.5-14B-Instruct and Qwen2.5-32B-Instruct, are asked, one item at a time and blind to the label, whether the reply was written by a person, and the summary is the AUC, the chance a judge ranks a real reply above a generated one, so 0.5 is chance.

At the level of a sentence the judges do not reliably tell the text from real. On the deployed configuration the AUC runs from 0.514 to 0.529 across two judges and two control samples and

every interval contains 0.5 (Table S5, middle block); the judges call 98–100% of generated first sentences human-written against 94–98% of real ones. The automatic statistics agree: 12.1 words against 14.1, type-token ratio 0.950 against 0.953, distinct-3 0.998 against 0.999.

Left to stop by itself, over a whole reply it is separable, at AUC 0.720 to 0.802, with 62–78% of generated replies called human-written. A generated reply averages 105.3 words against 36.0 for a real one, and the residue is repetition rather than topic. Against real replies of comparable length the generated ones score type-token ratio 0.650 against 0.765 and distinct-3 0.944 against 0.992, while lexical overlap with the parent stays in the same range. The factor writes a good sentence, and knows where a reply could end, and still ends it too late.

Learning where a reply ends. Where a reply ends belongs to the text factor of Equation 1, and it is learned like the rest of it. A linear quantile model at 101 levels reads the community and the lengths of the two posts the reply sits between (the post being replied to, and the actor's own previous post) and a target is drawn from its predicted distribution per event; the reply is then cut at the nearest sentence boundary. It is fitted on the training segment, on 99,994 replies, and scored out of sample on 19,999 it never saw.

We report it as an ablation over its own features (Table S4), ranked by how well the predicted length orders the realised one. Given only the community, the factor reproduces the table: $\rho = 0.268$ against the per-community median's 0.265, a paired difference of $+0.003$ [+0.001, +0.005]. What a reply's length depends on is the length of the post it answers. Adding that moves ρ to 0.392, and adding the actor's own previous post to 0.404, $+0.139$ [+0.127, +0.151] over the median. The table is therefore this factor restricted to the community, and nearly all of the factor's gain comes from the two posts the table cannot read. The simulator's own world-state summary z adds nothing on top (0.404 against 0.404) and predicts almost nothing alone (0.032).

One mechanical point changes every number below. Cutting at the last sentence boundary *at or below* a target loses length in one direction only, every time; cutting at the *nearest* reachable boundary does not. On 3,956 held-out replies that one change moves the emitted length distribution from a Wasserstein-1 distance $W_1 = 36.8$ characters from real to 20.5 on the per-community arm, with no fitted parameters, and it applies to every arm we report.

The factor closes most of what is left. Emitted length falls from 542 characters to 131 against a real 202, and replies ending on terminal punctuation rise from 72% to 96.6%. Whole-reply separability falls from AUC 0.720–0.802 to 0.562–0.664 and the share of generated replies judged human-written rises from 62–78% to 95–99%, against 94–100% for the real replies they are set beside (Table S5 and Figure S2).

The clearest way to read that is as a collapse of the gap between the two units. Left to stop by itself the whole reply was separable at up to 0.802 while its own first sentence sat between 0.559 and 0.614, a spread of about 0.24 AUC between two views of the same text, which points to length as the cause. With the factor the whole reply lands at 0.562–0.664 against a first sentence at 0.514–0.529: a spread of about 0.15, so the unit the system emits is now judged on roughly the same terms as the sentence it was always able to write.

Against the table it replaces, on the same 116 events. The bottom block of Table S5 is the same factor with its context features removed, generated in the same way and differing in nothing else. The learned factor is nominally ahead in all eight judge cells, and at $n = 116$ every interval is about 0.15 wide, so no single cell separates them; the powered comparison is a paired study on 3,956 prompts, where the two are a tie (ΔAUC $= +0.0002$ on the smaller judge and -0.0035 on the larger, against a minimum detectable effect of 0.011 and 0.008). To judges the two are the same. They are not the same on ordering, where the learned factor is ahead by $+0.059$ [+0.045, +0.075].

Conditional accuracy and aggregate fit are different targets. A factor fitted per event trades a little aggregate fit for conditional accuracy: $W_1 = 25.1$ characters from real against the per-community table's 20.5, a difference of $+4.6$ [2.7, 6.3].

Each stopped condition is a *separate generation run* rather than a truncation of the stored texts, so its blocks differ by sampling as well as by rule; that is why the first-sentence figures move at all when the first sentence is nominally the same object, and why their intervals overlap throughout.

Table S4: Feature ablation of the length factor. Out-of-sample rank correlation between predicted and realised reply length, fitted on the training segment (99,994 replies) and scored on 19,999 held out from it, with paired 95% intervals against the per-community median it replaces, over 4,000 bootstrap draws. Given only the community, which is all the table reads, the factor matches the table (+0.003); the length of the post being replied to is worth +0.092 on its own. The deployed configuration is in bold.

features the factor reads	ρ	vs. the median
per-community median (the table)	0.265	—
community	0.268	+0.003 [0.001, 0.005]
parent	0.357	+0.092 [0.076, 0.107]
parent + previous post	0.381	+0.116 [0.101, 0.131]
community + parent	0.392	+0.126 [0.116, 0.137]
community + parent + previous post	**0.404**	**+0.139** [0.127, 0.151]
world-state summary z	0.032	−0.234 [–0.253, –0.215]
all four	0.404	+0.139 [0.127, 0.151]

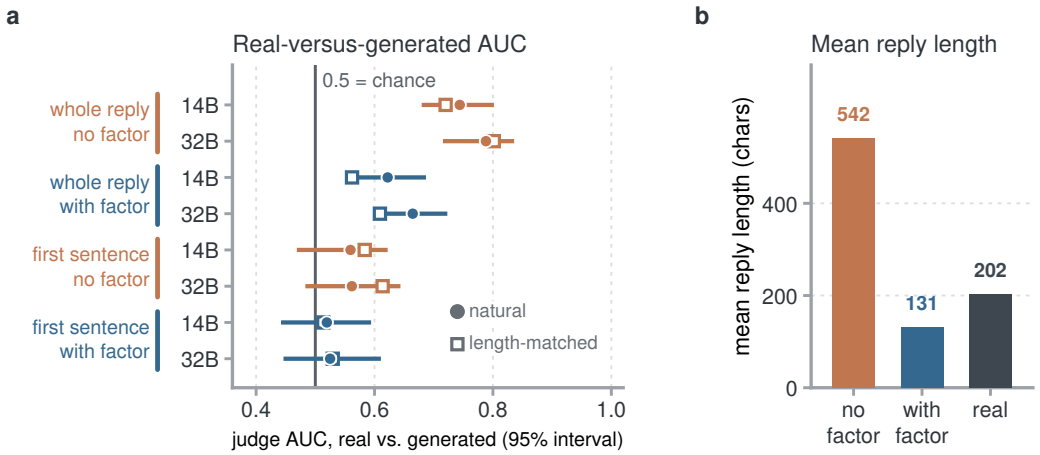

Figure S2: The first two blocks of Table S5 as a picture, on the deployed two-stage factor. **(a)** judge AUC against real controls, with and without the length factor, both judges, 0.5 marked. With the factor the whole reply is judged on roughly the same terms as its own first sentence; left to stop by itself it separates. **(b)** why: unaided the mean reply runs to 542 characters against a real 202, and the factor brings it to 131.

The repetition the length control exposed mostly closes with the factor. Against real replies of comparable length, a stopped reply scores type-token ratio 0.873 against 0.921 and distinct-3 0.994 against 0.999, where the unstopped factor scored 0.650 against 0.765 and 0.944 against 0.992. Most of the excess repetition was a function of running long rather than a property of the writing. A residue remains, the same size on both stopped arms, so it belongs to the writing and not to where the reply is cut. What the factor does *not* match is the length itself, 26.5 words against a real 36.0, undershooting where the unstopped factor overshot at 105.3.

E Recursive closure

Population-level fidelity says the simulator emits the right amount of the right kind of activity. It does not say the simulator is a population. That requires the events it generates to become the world the next events are generated from: a generated post has to be selectable as a later actor's reply target, and its text has to be what that actor reads. A system whose output only ever attaches to real history is conditioning on the past, however well it scores. This section evaluates that property directly, on the event graph.

The rate the corpus sets. Of the 90,637 replies real users wrote inside the five 24-hour held-out windows, 85.18% named a parent created inside that same window. The population overwhelmingly replies to what it has just said. Those in-window chains run to a mean depth of 3.285. A simulator at 0% and one at 100% are both wrong, and so is a system that hits the rate with the wrong shape: the

Table S5: Real-versus-generated discrimination on the 116 complete events, blind and one item at a time, scored on the two-stage text factor the simulator deploys. AUC is the chance a judge ranks a real reply above a generated one, so 0.5 is chance; brackets are 95% bootstrap intervals. *First sentence* ends where the model put a full stop. In the upper block *whole reply* ends where the factor's own stop token fires, which is 114/116 of the time; in the other two it ends where the length factor put it. The bottom block is that factor with its context features removed, which is numerically the per-community table it replaces; it is generated on the same events with the same adapter, sentence-boundary rule and token budget, so the pair is an ablation. Real controls are matched on community, window and parent age, and the second control additionally on length.

unit	judge	real control	AUC	generated called human
no length factor: the reply ends where the text factor's own stop token fires				
first sentence	Qwen2.5-14B	natural	0.559 [0.469, 0.622]	91%
first sentence	Qwen2.5-14B	length-matched	0.583 [0.503, 0.668]	91%
first sentence	Qwen2.5-32B	natural	0.562 [0.483, 0.644]	97%
first sentence	Qwen2.5-32B	length-matched	0.614 [0.545, 0.697]	97%
whole reply	Qwen2.5-14B	natural	0.744 [0.680, 0.802]	62%
whole reply	Qwen2.5-14B	length-matched	0.720 [0.646, 0.771]	62%
whole reply	Qwen2.5-32B	natural	0.788 [0.716, 0.836]	78%
whole reply	Qwen2.5-32B	length-matched	0.802 [0.716, 0.839]	78%
with the learned length factor: community, parent length, own previous-post length				
first sentence	Qwen2.5-14B	natural	0.519 [0.442, 0.594]	98%
first sentence	Qwen2.5-14B	length-matched	0.514 [0.437, 0.598]	98%
first sentence	Qwen2.5-32B	natural	0.525 [0.446, 0.611]	100%
first sentence	Qwen2.5-32B	length-matched	0.529 [0.452, 0.598]	100%
whole reply	Qwen2.5-14B	natural	**0.622** [0.559, 0.687]	**95**%
whole reply	Qwen2.5-14B	length-matched	**0.562** [0.492, 0.622]	**95**%
whole reply	Qwen2.5-32B	natural	**0.664** [0.604, 0.723]	**99**%
whole reply	Qwen2.5-32B	length-matched	**0.609** [0.534, 0.691]	**99**%
the same factor with its context features removed, the per-community distribution				
first sentence	Qwen2.5-14B	natural	0.554 [0.478, 0.636]	96%
first sentence	Qwen2.5-14B	length-matched	0.577 [0.492, 0.651]	96%
first sentence	Qwen2.5-32B	natural	0.546 [0.479, 0.620]	97%
first sentence	Qwen2.5-32B	length-matched	0.564 [0.490, 0.638]	97%
whole reply	Qwen2.5-14B	natural	0.660 [0.588, 0.736]	89%
whole reply	Qwen2.5-14B	length-matched	0.601 [0.525, 0.681]	89%
whole reply	Qwen2.5-32B	natural	0.690 [0.618, 0.756]	96%
whole reply	Qwen2.5-32B	length-matched	0.654 [0.578, 0.726]	96%

6 h trailing-window protocol replayed on these windows attaches to a generated post 79.26% of the time while emitting 14.14 posts per actor against a real 1.91. Both the rate and the activity it rides on are what an attachment factor is judged on.

Table S6: Recursive closure on the event graph, five held-out windows and three seeds (15 pairs per arm, 96 steps). *In-window parent* is the share of replies whose parent was created inside the window (for the simulator, a generated one); *depth* is the mean depth in the window's own reply forest, where depth 1 is attachment to a parent from before the window. The *narrow-pool* control is the target head fitted on 6 h candidate pools while the simulator serves it 7 d ones; it is a training-pool mismatch, not the 6 h trailing-window protocol of §5.1. The last rows give the attachment factor's effect on the two fidelity axes of §6.

actor scorer	reply target	in-window parent	depth
hazard head	narrow-pool target-head control	0.00%	1.000
attention	served-pool head, **U** controller	82.07%	4.195
attention	narrow-pool target-head control	0.00%	1.000
attention	**served-pool target head**	**82.35%**	**4.243**
real corpus		85.18%	3.285
effect on the fidelity axes, paired, 95% over (seed, window)			
Δ coverage, served-pool − narrow-pool		−0.0046 [−0.0060, −0.0032]	
Δ precision, served-pool − narrow-pool		−0.0028 [−0.0042, −0.0016]	

The deployed model reproduces the corpus rate (Table S6): 82.35% of its replies name a generated parent against 85.18% in real data, at a mean chain depth of 4.243 against 3.285.

Closure with text. If a generated post is a reply target then its text is what the replying actor reads, so the property can only be demonstrated end to end. We first ran the structural factors of the deployed model: predicted rate, composition allocation, attention scorer, multiplicity table and served-pool target head. They emit 2,494,058 events in 440,145 reply trees, 2,053,913 of them answering a post the same run generated. The unit of work is the generated tree, so that a parent and its child are always in the same shard; since the simulator emits at $t + U[0, \Delta)$ while drawing candidates from $[t - W, t)$, timestamp order inside a tree is a topological order. Trees are drawn *uniformly at random* from the 440,145 until the event budget is met, not chosen for depth (selecting deep trees would inflate the statistic this run exists to report).

We then ran the learned length factor and the two-stage text factor on the 413 trees drawn, writing all 1,575 of their events in full. Of these, 1,162 (73.8%) reply to a generated post whose text this run produced. The chains reach depth 11: a generated actor replying to a generated post, whose author was replying to a generated post, ten levels down to a real one.

Table S7: The full-event run, 1,575 complete events on the held-out segment, stratified by what the actor was replying to and by depth in the generated forest. *Distinct-3* is the within-reply distinct trigram ratio; higher is less repetitive.

	n	mean chars	ends on punct.	distinct-3
parent is real	413	169.1	98.1%	0.9819
parent is generated	1,162	164.5	97.7%	0.9825
depth 1	413	169.1	98.1%	0.9819
depth 2–3	592	164.0	97.8%	0.9826
depth 4+	570	165.0	97.5%	0.9824

Sample size. The comparison that matters is depth 1 against depth 4+, where a reply follows at least three generated events up its chain: 413 and 570 events. At those sizes a two-sided test at 80% power detects a 1.76% change in distinct-3 and a 15.9% change in mean length; the observed differences are +0.05% and −2.4%. The run therefore resolves a change in *within-reply* diversity down to under two percent, and a change in length only down to about a sixth. Trees are the sampling unit because a parent must be written before its child, and they are drawn uniformly, so the tier is a random sample of the forest's trees rather than of its events. Small trees are numerous, shallow events are over-represented relative to the structural forest, and the depth-4+ group is the smaller for it.

Two kinds of diversity. Distinct-3 is a *within-reply* quantity: how much one reply repeats itself. What recursive generation is known to lose is the other kind, outputs converging on one another (Alemohammad et al., 2024; Shumailov et al., 2024), and that is measured *across* replies, not inside them. The deep group has not converged. At depth 4 and beyond, 567 of the 570 replies are distinct strings; the median token Jaccard between two of them is 0.0597 against 0.0577 at depth 1, so they are only slightly more alike than replies written directly onto real posts. Their overlap with the parent they answer is *higher*, 0.1411 against 0.1125, so four or more levels into a generated chain a reply is still about the post above it rather than drifting into generic text.

Whatever the chain accumulates, the text shows no detectable degeneration (Table S7). Table S8 gives every event one actor emitted, verbatim.

Why the attachment factor is fitted on the pool it is served. The reply-target head reads ten features of each candidate, and the pool it scores at serving time spans a seven-day window in which generated posts reach 12.09% of the candidates by the last step, so what a head does with them is a question about ranking rather than about availability. Fitting it on a narrower pool leaves it extrapolating: on a six-hour training pool the real candidates at serving carry *since last touch* at $z = +3.22$ against the training mean, and past the edge of its support a network's ranking is whatever its last linear region does. That verdict lands on the out-of-range real nodes, which are most of the pool, and swamps every candidate inside it, which is all of the generated ones. The failure is one of support: the narrow-pool target-head control selects a generated parent 0 times in 2,869,085 draws. Scoring it on pools replayed from the simulator's own recorded state puts generated candidates at percentile 0.063, and neutralising *since last touch* alone lifts them to 0.384 while every other feature leaves them within 0.001 of where they started (Figure S3). The same range effect selects the nodes

Table S8: Every event one actor emitted inside one held-out window of the run of Table S7, verbatim. Of the actors emitting three or more events in a window, this is the one whose events span the most steps, 15 to 58 of 96, which is what the table is here to show: the loop persists across the rollout rather than inside a single step. The reply target is chosen by the model; the minute within a step is the uniform draw of Equation 1, and the community follows from the parent. Depth is position in the reply chain. Two of the three answer a post this run wrote, and the last is also prompted with one of its own. Account handles are pseudonymised and the corpus is public.

1st step 15	`r/relationship_advice`, depth 3, *replying to a post this run generated*: "You should have a discussion about her income to be able to support both of you. Is it possible for her to not have the income in the future?" *prompted with a real post of its own* → "And is she able to support herself? If not, why would she not work? The OP said she's not looking for jobs right now, because she doesn't have any experience, and even if she worked, the money wouldn't stretch to cover bills."
2nd step 19	`r/relationship_advice`, depth 1, *replying to a real post* [not reproduced; about who should be best man] *prompted with a real post of its own* → "Or just two best men. You and your son. One who's a man and one who's a woman. A traditional best man and a new best man. Who's the best man? The guy that fits."
3rd step 58	`r/relationship_advice`, depth 5, *replying to a post this run generated*: "This is the way of the world. She's not being cruel, she's trying to do what she's told." *prompted with a generated post of its own* → "She's being cruel, to be honest. She should have been told that he was in a relationship with a 20 year old before. That would have saved her some heartache. Why can't she at least accept his explanation?"

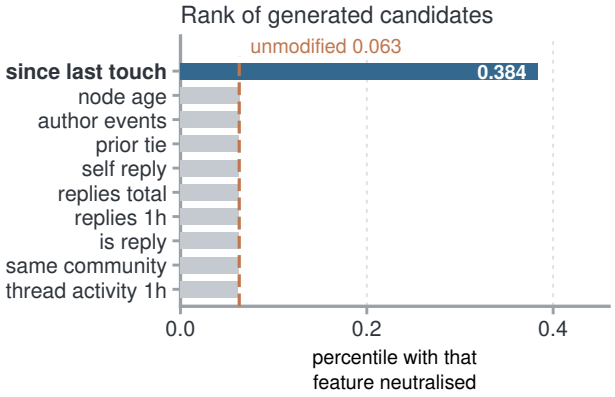

Figure S3: Where the narrow-pool control ranks generated candidates once the named feature is set to the real candidates' median, on pools replayed from validation rollouts; the orange line is the unmodified rank. Only *since last touch* moves them: a support failure, not a preference against generated posts.

least likely to attract a reply. Such a head chose $17{,}524$ distinct real parents of which 1.86% ever received a real reply, against 43.68% for a control matched on reply type and on age in one-day bins and 34.10% for an unmatched draw from the same pool.

Fitted instead on candidate pools drawn by the simulator's own rule at the window it serves, the identical architecture, features and recipe come close to the corpus rate. This is the lesson §I.4 reports for the actor head, where training on the served distribution was worth about four times the choice of scorer, applied to the factor that needed it more.

The control isolates the cause (Table S9). Refitting on the original six-hour pool with the same recipe reproduces the narrow-pool target-head control's behaviour to three decimal places (generated percentile 0.0632 against 0.0631, and the same zero), so the difference comes from the pool. On the served pool the same architecture goes from $24\times$ worse than uniform to less than half of uniform, and from 0% to 87.87% attachment on replayed pools. Two additions do not pay for themselves. A per-feature piecewise-linear encoding, which cannot extrapolate by construction, overshoots to 93.72% and costs held-out likelihood (once the support is matched there is no extrapolation left to guard

Table S9: Reply-target head, three seeds. *Validation* is top-1 accuracy and per-episode NLL on the 20,000 validation episodes of Appendix L, whose candidate pools are drawn the way the simulator draws them; uniform over 1024 candidates is NLL 6.931. *In the loop* is measured on pools replayed from the simulator's recorded world state, so every arm is scored on identical pools.

		validation		in the loop	
	candidate pool	top-1	NLL	gen. percentile	attaches to generated
narrow-pool target-head control	6 h (as trained)	—	—	0.0631	0.00%
control refit	6 h	0.0066	163.41	0.0632	0.00%
served-pool target head	**7 d, as served**	**0.2989**	**3.17**	**0.9565**	**87.87%**
+ quantile encoding	7 d	0.2690	3.27	0.9543	93.72%
+ encoding + replyability	7 d	0.2698	3.26	—	—
same, on permuted labels		0.2570	3.29	—	—

against), and an auxiliary head predicting whether a candidate is replied to at all carries real signal over permuted labels, but not enough to overtake. We serve the plain served-pool target head.

F Burstiness and recency composition

Activity is bursty; there is no refractory period. Given that a user acted in step k, the marginal hazard of acting at step $k + j$, relative to that user's own base rate, is 12.48 at $j = 1$, 7.61 at $j = 2$, 4.85 at $j = 4$, 3.22 at $j = 8$, 1.54 at $j = 24$ and never falls below 1.076 anywhere within 24 hours. Any mechanism that suppresses a just-active user therefore compensates for a ranking artefact rather than modelling a behavioural one, and §H names it accordingly.

The real recency composition is stable. Of the actors in a step who are reachable inside a seven-day window, the shares by time since previous post are $m = (0.3165, 0.1701, 0.1888, 0.3246)$ on the training segment, and the freshest share varies only between 0.28 and 0.33 across the 24 hours of the day. A single fixed composition is therefore an adequate constraint, and we use the measured one.

G The global rate model

Let D_t be the number of reachable distinct actors in step t (actors excluding never-seen births, per §2.2). We predict it with a log-linear model

$$\log \hat{D}_t = s(\text{clock}_t) + \ell, \qquad \ell = \log\Big(\textstyle\sum_{t' \in P} D_{t'} \Big/ \sum_{t' \in P} e^{s(\text{clock}_{t'})} \Big), \tag{2}$$

where s is a Poisson GLM on hour-of-day harmonics ($k \le 4$) and day-of-week indicators, fit on the training segment and frozen, and ℓ is a *single scalar* calibrated on the prefix window $P = [t_{\text{start}} - 28\,\text{d},\ t_{\text{start}})$: four whole weeks of real events strictly before the rollout begins.

Two properties matter.

It is not an oracle. P ends strictly before t_{start}. It is the same visible prefix every actor feature already reads. What arm **V** reads, and what we remove, is the real actor count *at each step inside* the rollout.

Its loop gain is exactly zero. s is a function of t alone and ℓ is fixed before the first generated event exists, so $\partial \log \hat{D}_t / \partial(\text{anything emitted}) = 0$ identically.

Choosing a clock-only rate model is a deliberate trade. Feature sets that read the running event counts explain far more of the step-level variance (out-of-sample R^2 of 0.9215 against 0.3456 for clock-only) but their loop gain $\gamma = \partial \log \hat{D}_t / \partial \log(\text{emitted volume})$ is 0.970 and 0.904, i.e. barely contracting, and a simulator whose total is 97% self-reinforcing is one perturbation away from divergence. We take the loop-free option and pay for it in per-step accuracy: the clock-only model's per-step relative error is 8.8%, and on six held-out windows spanning 2022-02 to 2022-12, seven to seventeen months past the 2021-06-30 end of its training segment, it predicts the reachable-actor count at 0.9873 ± 0.0480. §6.3 shows that the accuracy given up costs nothing downstream.

H Composition allocation

Given $\hat{D}_t$, we split it across the four recency strata of §F in their measured real proportions m by largest remainder, capped by stratum occupancy, with any capped mass redistributed over the others. The composition of the emitted population is then correct by construction, and, because the mass per stratum is fixed, anything we do *inside* a stratum cannot distort it.

Inside each stratum we select by mass-conserving Bernoulli sampling. Let s_u be the hazard logit for candidate u and D_c the mass assigned to stratum c. We solve for the scalar offset b_c with

$$\sum_{u \in c} \sigma\Big(\frac{s_u}{\tau} - \rho \mathbb{1}[u \text{ selected within } K \text{ steps}] + b_c\Big) = D_c \tag{3}$$

by bisection, and draw independent Bernoullis. Three things are worth stating about Equation 3.

The bisection is the stabiliser. However far s_u drifts as the simulated world diverges from the real one, the expected count in stratum c is exactly D_c. This is what allows us to reuse a frozen hazard head whose absolute calibration is not guaranteed to survive 96 closed-loop steps: only its logit differences enter. It is also what removes the failure mode that a per-user-hazard formulation has: there, $\sum_u \sigma(s_u)$ is accurate to 4% at step 0 yet reaches $16.4\times$ real volume by step 24, because the simulator's own output inflates the features the hazard reads.

τ is a selection temperature. At $\tau = 1$ the draw keeps the head's own logit differences; smaller τ sharpens towards top-k. It trades coverage against precision at constant volume, because the bisection re-solves.

ρ is a repeat-selection penalty. It subtracts ρ logits from candidates selected within the previous K steps. §F measured that real activity is bursty at every lag, so this term does not model a behavioural mechanism; it compensates for a ranking that concentrates more than real volume does. We apply it *inside* strata precisely so that it cannot distort the composition. §I.3 gives its measured effect.

Each selected actor then emits $n \sim P(n \mid \text{lifetime-rate bucket})$ events from a frozen 12×8 empirical multiplicity table measured on the training segment (overall mean 1.245).

I Design decisions

This section gives the evidence for the design choices in §5, on validation unless a subsection says otherwise.

I.1 Resampling and temperature

Each emitted event picks a parent from a pool of 1024 candidates drawn from `unique(nodes created in` $[t - 7\text{d}, t)$ `∪ parents of events in that window)`. The frozen head B was trained by cross-entropy over exactly such a 1024-candidate set, sampled independently *per event*. Serving must match that in two respects, and the trailing-window construction matches neither:

1. **Sample, do not sharpen.** $\arg\max_j(\text{score}_j + \text{Gumbel}(0,1))$ is exactly a draw from $\text{softmax}(\text{score})$. Using a Gumbel scale of g samples from $\text{softmax}(\text{score}/g)$, i.e. at temperature g. We use $g = 1.0$.
2. **Resample the pool.** Training drew a fresh pool per event; sharing one pool across all the actors in a step concentrates the whole step's replies on whichever few nodes score highest. We draw a fresh pool every 32 actors.

Temperature alone cannot fix it (Table S10). Raising the Gumbel scale from 0.3 to 4.0 only reaches degree $W_1 = 2.95$. Restoring the per-event pool resampling that training used takes it to 0.34 at 32 actors and 0.12 at 16. We use 32 as the compromise between fidelity and the linear cost in feature computation. Together the two move the number of distinct parents per step from 24 to 909, against a real 704, and precision *rises* 15% relative to the prior setting.

Table S10: Attachment decoder, two windows of validation. *degree* W_1 is the Wasserstein-1 distance between the simulated and real distributions of replies per parent; the real data spreads a step's replies over 704 distinct parents.

setting	degree W_1	distinct parents / step	precision
Gumbel scale 0.3, one pool per step (arm **V**)	22.04	24	0.0155
Gumbel scale 4.0, one pool per step	2.95	353	0.0172
scale 1.0, fresh pool every 256 actors	2.87	359	0.0186
scale 1.0, fresh pool every 32 actors	**0.34**	**909**	**0.0178**
scale 1.0, fresh pool every 16 actors	0.12	1120	0.0177

I.2 The dormant-user channel

26.86% of real next-step actors have been silent for over a week, so a dormant channel is the obvious way to reach them. We implemented one, a stratified return hazard over 288 (gap, lifetime-rate) cells with exact uniform sampling inside a cell, and measured it as a net negative. Switching it on moved coverage from 0.1578 to 0.1418 and precision from 0.0178 to 0.0117. The dormant tier holds $\sim$11.5M users at a measured return rate of 3.0×10^{-5} per step; mass placed there draws at two orders of magnitude below the 2.4×10^{-3} available in the active tier, and no stratum weighting recovers that. So the channel is off, and the dormant population stays unreachable by design.

I.3 Composition pinning

Two measurements justify Equation 3's structure.

Pinning the composition is a no-op at the calibrated point. At $\tau = 1$ with no penalty, imposing the measured composition changes nothing: coverage $0.1578 \rightarrow 0.1578$, precision $0.0178 \rightarrow 0.0180$. The hazard is already calibrated *across* recency strata. The constraint is there to protect the composition once τ and ρ start moving mass.

The penalty moves the frontier, it does not slide along it. Judged on precision alone the penalty looks harmful. Judged at matched coverage it is the largest single gain we found: at coverage 0.1717 the penalised configuration scores precision 0.0301, against 0.0115 for the temperature-only frontier interpolated to the same coverage, a factor of 2.63. Figure 5b places the five held-out arms against this sweep.

I.4 Training distribution and scorer class

§B reports that a gradient-boosted forest on the same features beats the trained hazard by up to 0.026 AUC in the largest stratum. That invites the control the measurement asks for: serve the forest instead of the head and change nothing else. Both arms here are superseded by the attention scorer of §I.5, which the simulator deploys; what this section establishes is the ordering between training distribution and model class.

The forests are fitted per stratum, which matches both that measurement's construction and the controller's own, since selection already happens within strata; their probabilities are converted to logits so the bisection of Equation 3 solves an offset on the same scale. Everything else is the treatment's: the predicted total, the strata, τ, ρ, the composition constraint, the attachment decoder, the frozen actor score that both scorers read, and the windows and seeds. The forest is fitted in two versions, because the head carries an advantage that is easy to overlook: one round of dataset aggregation on drifted features (§C). A forest fitted on real histories alone is one round short of it.

The result (Table S11) splits cleanly. Fitted on real histories alone the forest is the weaker in-loop scorer on coverage by a wide margin (0.1697 against 0.1921) and it gets there while emitting 1,184 *more* distinct actors, so the deficit is ranking rather than volume. Give it the same drifted rows the head was given and it beats the head on both axes: $+0.0080$ coverage and $+0.0044$ precision, both intervals clear of zero. Precision is per selection, so that half is ranking outright; of the 4.1% coverage gap, 1.7 points is the extra actors it emits and the rest is not.

Table S11: Serving a per-stratum forest in place of the hazard head, on validation. All figures are on the same nine (seed, window) pairs; the head arm is the validation configuration of §5, identical to the treatment in every controller setting but carrying the hazard head rather than the attention scorer the simulator deploys. Bracketed values are 95% bootstrap intervals, and every one of them is unchanged in sign under window clustering.

actor scorer	coverage	precision	volume	distinct actors
hazard head, one DAgger round	0.1921	0.0313	1.008	50,539
per-stratum forest, real histories only	0.1697	0.0329	1.019	51,723
per-stratum forest, same drifted rows	**0.2001**	**0.0357**	1.022	51,392
paired contrasts				
head − forest (real histories only)	+0.0224 [+0.0195, +0.0249]	−0.0016 [−0.0023, −0.0008]		
head − forest (same drifted rows)	−0.0080 [−0.0092, −0.0066]	−0.0044 [−0.0049, −0.0039]		
forest with − without drifted rows	+0.0304 [+0.0276, +0.0331]	+0.0028 [+0.0018, +0.0036]		

The larger number in that table is the third row. Pooling the drifted rows is worth $+0.0304$ coverage to the forest, against $+0.0080$ for the change of model class at matched data, so **training the scorer on the distribution it will be served matters about four times more than what the scorer is**. That is the same conclusion §C reaches for the head, measured again on a learner that shares none of its inductive bias.

I.5 The attention scorer

The forest result is a statement about model class at fixed features, and it invites the question behind it: is the gap a property of trees, or of what the neural head was given to read? An ensemble would settle the score without answering that, so the search is over single models. It ran on a training-segment holdout, where the held-out segment plays no part, and is scored by AUC weighted by the recency composition the controller is actually pinned to, since a scorer that wins only where the controller rarely selects is not a better scorer for this controller. The search ran seventy-three fits, starting from the shared-MLP hazard baseline's own shape and adding one mechanism at a time (Table S12).

Table S12: Actor-scorer ablation on a training-segment holdout, one seed. The metric is AUC weighted by the emitted recency composition, and Figure S4 plots it. *Gap closed* is the fraction of the distance from the shared MLP to the per-stratum forest ($0.7843 \to 0.7893$) that each variant covers, so 100% is the forest and above it is past the forest. Every row trains for the same 24,000 steps: at 8,000 the shared MLP scores 0.7778.

actor scorer	weighted AUC	gap closed	params
shared MLP (control)	0.7843	0%	404k
+ loss normalised within each stratum	0.7769	−149%	404k
+ hazard baseline in log gap plus ranking residual	0.7848	9%	404k
+ soft mixture of experts, gap-dependent gate	0.7857	27%	1,616k
+ four hard per-stratum output heads	0.7859	33%	406k
+ FiLM, every layer modulated by a basis in log gap	0.7862	39%	439k
+ per-feature piecewise-linear encoding, 64 bins	0.7883	80%	985k
+ encoding and gap-conditional standardisation	0.7885	84%	985k
+ attention over per-feature tokens, *no* encoding	0.7886	86%	**362**k
+ attention over per-feature tokens *with* the encoding	**0.7913**	**142**%	369k
reference and rejected variants			
per-stratum forest, same drifted rows (the target)	0.7893	100%	—
deployed model + disjoint-band counts and rates	0.7915	145%	369k
encoding, distilled from the seven-member ensemble	0.7899	113%	985k

Two mechanisms carry it, and neither is capacity. Giving each numeric feature a **piecewise-linear encoding** rather than a scalar, a learned embedding over 64 quantile bins (Gorishniy et al., 2022), closes 80% of the gap on its own, against the five architectural changes above it in the table, the best of which reaches 39%. Treating the features as a **set of tokens and letting them attend to each other**, in the manner of a feature tokeniser (Gorishniy et al., 2021), closes 86% with 362k parameters, fewer than the shared-MLP baseline's 404k. The two are not the same lever: composed, they reach 0.7913, past the forest rather than level with it, at 369k parameters. Normalising the loss within each stratum is actively harmful. A mixture of experts buys a quarter of the gap for four times the weights. And distilling an ensemble of seven independently seeded copies of the encoding model into one

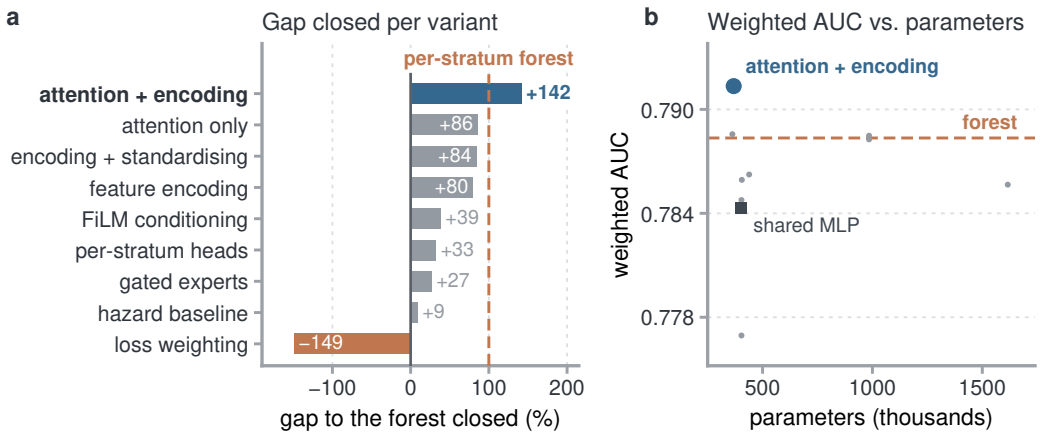

Figure S4: The actor-scorer search, on a training-segment holdout. **(a)** each variant as a share of the distance from the shared MLP to the per-stratum forest, one mechanism at a time and every one trained to the same 24,000 steps. Only the row that composes both mechanisms clears the forest. **(b)** the same fits against parameter count. The winner is not the largest model: it is smaller than the shared-MLP baseline, and the mixture of experts spends four times the weights to reach a quarter of the gap.

student lands at 113%: real, and below the 142% that training the single model directly reaches, so the ensemble is not holding anything the architecture cannot.

The comparison, with the inputs held identical. A scorer can win by reading more, so the main comparison fixes that. Both sides get the same 18 features and the same 256 context dimensions, are fitted on the same drifted rows, and are scored on the same holdout episodes; the only difference is the model. Across three seeds the attention scorer beats the forest by $+0.0030$, $+0.0032$ and $+0.0032$, mean $+0.0031$, with the same sign in all four strata on all three seeds. Read against the stronger forest variant instead, the one given only the 16 context dimensions it was originally fitted with, the margin is $+0.0023$. Disjoint-band counts and rates are worth $+0.0002$ at full training, inside seed noise, so they are off.

I.6 The scorer substitution

Everything in §I.4 and §I.5 is validation evidence. The forest was compared to the head on validation, and the architecture was selected on a training-segment holdout. Neither is held out in the sense §6 means. So we ran the substitution as its own held-out pass, on the held-out segment, on windows disjoint from those §6 uses.

The scorers, baselines, window rule and analysis were all fixed before any of these windows was read.

Windows, and why they are disjoint. Both sets come from one 13-point grid laid over the held-out segment by the same rule. §6 evaluates the even indices $2, 4, 6, 8, 10$; this pass evaluates the odd indices $1, 3, 5, 7, 9, 11$. The two are disjoint by parity, and no window of either set falls within 24 hours of a window of the other. They span 2022-02 to 2022-12.

Arms. Three, differing *only* in the actor scorer: the hazard head, the per-stratum forest on the same drifted rows, and the attention scorer of §I.5. Every other component is the treatment's: the predicted total, the composition allocation, τ, ρ, the mass-conserving bisection, the attachment decoder, the multiplicity table and the frozen actor score all three read. Three seeds $\times$ six windows $\times$ three arms is 54 single-window rollouts.

What it found. The ordering seen on validation, the attention scorer ahead of both, holds on all four contrasts of Table S13, and every interval is clear of zero under both the (seed, window) bootstrap and the window-clustered one. Against the hazard head the attention scorer is $+0.0162$ coverage and $+0.0054$ precision; against the forest, $+0.0103$ and $+0.0034$. The forest in turn beats the head, which is the §I.4 result reproducing out of sample.

Table S13: The scorer substitution, on six held-out windows disjoint from the five §6 uses. All figures are on the same 18 (seed, window) pairs. Brackets are 95% bootstrap intervals over (seed, window); the window-clustered interval is given for every contrast because six windows, not 18 pairs, are the replication unit.

actor scorer	coverage	precision	volume	posts/actor	distinct actors
hazard head (MLP)	0.1885 [0.1837, 0.1936]	0.0313 [0.0304, 0.0322]	0.955	2.93	55,287
per-stratum forest, same data	0.1944 [0.1902, 0.1988]	0.0334 [0.0321, 0.0346]	0.963	2.91	56,157
attention scorer	**0.2047** [0.2002, 0.2095]	**0.0367** [0.0357, 0.0377]	0.958	2.83	57,325

contrast	quantity	Δ	95% CI	clustered on window
attention − head	coverage	+0.0162	[+0.0154, +0.0171]	[+0.0154, +0.0173]
attention − head	precision	+0.0054	[+0.0050, +0.0057]	[+0.0049, +0.0058]
attention − forest	coverage	+0.0103	[+0.0081, +0.0125]	[+0.0068, +0.0135]
attention − forest	precision	+0.0034	[+0.0025, +0.0042]	[+0.0022, +0.0046]
forest − head	coverage	+0.0059	[+0.0040, +0.0081]	[+0.0030, +0.0094]
forest − head	precision	+0.0020	[+0.0011, +0.0029]	[+0.0007, +0.0030]

Coverage counts distinct actors, so an arm that simply emits more of them can score higher without ranking better; precision is per selection and is immune to that. The decomposition says the gain is mostly not volume. Set against the attention scorer, the head emits 3.6% fewer distinct actors but takes 7.9% less coverage. Had the gap been volume alone those two shortfalls would match, so under half of it is what the extra actors buy. Against the forest the same pair is 2.0% and 5.0%. The precision gains are ranking outright, since precision is per selection and all three arms hold volume to 0.955–0.963.

Where it is not uniform. The precision gain is concentrated in the freshest stratum. Per stratum, attention minus head is $+0.0160$, $+0.0006$, -0.0005 and $+0.0010$ across <1 h, 1–6 h, 6–24 h and 1–7 d; against the forest it is $+0.0110$, -0.0007, -0.0005 and $+0.0003$.

Scope of this pass. The six scorer-comparison windows are disjoint from the five primary windows but come from the same temporal segment, making this a within-corpus replication. The emitted recency composition stays within 0.30% of its target on every arm.

J Arms, paired contrasts and controls

Table S14 defines the five arms of §6.1 and Table S15 gives every paired contrast behind Table 4. A candidate pool built as the authors of a trailing window appears outside the LLM-agent literature in Mubang & Hall (2022), at a day's width and with new users injected each step, so Proposition 1 binds its returning-user branch only.

J.1 Additional controls

The controller's three settings contribute a real part of the result. **U** runs at $\tau = 1$, with no repeat-selection penalty and the composition left free, keeping every other setting of the treatment including the attachment resample rate, and still reaches coverage 0.1781 and precision 0.0174. Turning the three on adds $+0.0215$ coverage [+0.0129, +0.0295] and $+0.0170$ precision [+0.0159, +0.0181], i.e. $+12.1\%$ and $+97.8\%$ in relative terms, both intervals clear of zero. Two of the three, τ and (ρ, K), were chosen on validation; §I gives the evidence for each.

Removing the oracle costs nothing measurable. On the validation windows, using the predicted total instead of the *real* per-step reachable-actor count moved coverage by $+0.0004$ [–0.0019, +0.0029] and precision by $+0.0001$ [–0.0001, +0.0004], two ties. A clock-only rate with a prefix level is indistinguishable from reading the answer.

Stability. No arm diverged. Over the treatment's 1440 steps the worst single step emitted $1.0638\times$ real volume and the step-96 ratio was 0.9941.

Table S14: Arms, as the settings each was run with. **T** is the method of §5. **T**, **U** and **V** all read the same trained actor scorer; the two nulls replace it, which is the contrast they exist for. The pool is 7 d everywhere except **V**, whose width the construction it reproduces specifies.

id	name	what it is
T	ours (treatment)	§5 in full: $\tau = 0.4$, $\rho = 6$ logits over $K = 24$ steps, composition pinned, attachment pool redrawn every 32 actors at Gumbel scale 1.0, predicted total, multiplicity table, one DAgger round
U	no selection controls	identical, but $\tau = 1$, no penalty and composition free. Every other setting, the attachment resample rate included, is the treatment's
$\mathbf{N_r}$	null: rate	score $\propto$ lifetime post count. Same total, strata, τ, penalty, composition constraint, attachment decoder and reply-target head
$\mathbf{N_u}$	null: uniform	uniform score, otherwise identical to $\mathbf{N_r}$
V	6 h protocol	the trailing-window construction, driven by *our* actor scorer: 6 h actor pool and 6 h attachment window, the *real* per-step actor count as the budget, one event per actor in place of the multiplicity table, and Gumbel scale 0.3 with one pool per step in place of the attachment decoder

Table S15: Paired contrasts against the treatment under three estimators of the same Δ. The first resamples the fifteen (seed, window) pairs, the second resamples the five windows, and the third is a paired t on the five window means at 4 degrees of freedom, since a bootstrap over five clusters is anti-conservative.

contrast	quantity	Δ	95% CI, pairs	clustered on window	paired t, 5 windows
$\mathbf{T} - \mathbf{V}$	coverage	$+0.1527$	[+0.1434, +0.1611]	[+0.1355, +0.1656]	[+0.1290, +0.1764]
$\mathbf{T} - \mathbf{V}$	precision	$+0.0035$	[+0.0018, +0.0054]	[+0.0013, +0.0067]	[−0.0010, +0.0080]
$\mathbf{T} - \mathbf{N_r}$	coverage	$+0.0428$	[+0.0401, +0.0455]	[+0.0388, +0.0460]	[+0.0371, +0.0485]
$\mathbf{T} - \mathbf{N_r}$	precision	$+0.0141$	[+0.0129, +0.0151]	[+0.0133, +0.0148]	[+0.0129, +0.0152]
$\mathbf{T} - \mathbf{N_u}$	coverage	$+0.0917$	[+0.0874, +0.0957]	[+0.0849, +0.0977]	[+0.0814, +0.1019]
$\mathbf{T} - \mathbf{N_u}$	precision	$+0.0303$	[+0.0291, +0.0314]	[+0.0293, +0.0311]	[+0.0289, +0.0317]
$\mathbf{T} - \mathbf{U}$	coverage	$+0.0215$	[+0.0129, +0.0295]	[+0.0189, +0.0251]	[+0.0164, +0.0267]
$\mathbf{T} - \mathbf{U}$	precision	$+0.0170$	[+0.0159, +0.0181]	[+0.0159, +0.0179]	[+0.0154, +0.0187]

J.2 Per-window results

The replication unit is the window, so Table S16 gives each of the five. The aggregate is not carried by a window: the treatment is ahead of every arm in Table 4 on both axes in each of the five windows, and its weakest window reaches coverage 0.1787, still $2.11\times$ the mean 6 h closed-loop ceiling. Between-window variation is real and larger than between-seed variation (the treatment's coverage runs from 0.1787 to 0.2215 across windows while its widest seed spread inside any one window is 0.0089), which is why every interval in this work is also reported clustered on the window.

The narrowest margin in Table 4 is the precision contrast against the trailing-window protocol, $+0.0035$. Per window the treatment is ahead by $+0.0013$, $+0.0008$, $+0.0038$, $+0.0021$ and $+0.0097$, positive in each.

K Prompted models on the same task

K.1 Setup and scope of the comparison

Each of 120 episodes fixes the same 30 real preceding events, the same 20 candidate users and the same 10 candidate reply targets. The prompted models receive them as a text rendering: handles, posting history, last post and target text; our selector receives the same episode through its own feature builder. The episode, the candidate set and the question are identical across arms; the input format differs by model class. The question is the full event tuple: how many of the 20 act in the next 15 minutes, which, at what minute, replying to which target, and what they write.

Two properties of the episode sampler matter for reading the numbers. Positives are planted. A uniform 20-candidate sample from the active pool contains a real next-step actor with probability

Table S16: The five held-out windows one at a time, each the mean of three seeds. The right-hand column is the pooled figure Table 4 reports, which is the mean of the fifteen (seed, window) pairs and therefore of these five entries.

	window					pooled
arm	2	4	6	8	10	(Table 4)
distinct-actor coverage						
T, ours	**0.2019**	**0.2215**	**0.1979**	**0.1787**	**0.1980**	**0.1996**
U, no selection controls	0.1806	0.1930	0.1794	0.1577	0.1798	0.1781
$\mathbf{N_r}$, null: rate	0.1592	0.1757	0.1507	0.1434	0.1550	0.1568
$\mathbf{N_u}$, null: uniform	0.1106	0.1212	0.1005	0.0998	0.1077	0.1079
V, 6 h protocol	0.0569	0.0547	0.0336	0.0562	0.0331	0.0469
per-selection precision						
T, ours	**0.0338**	**0.0354**	**0.0349**	**0.0332**	**0.0351**	**0.0345**
U, no selection controls	0.0172	0.0177	0.0168	0.0183	0.0171	0.0174
$\mathbf{N_r}$, null: rate	0.0200	0.0201	0.0203	0.0205	0.0210	0.0204
$\mathbf{N_u}$, null: uniform	0.0043	0.0045	0.0036	0.0046	0.0040	0.0042
V, 6 h protocol	0.0325	0.0346	0.0310	0.0311	0.0254	0.0309

Table S17: Prompted-capability demonstration, 120 episodes. *lift* is actor precision divided by the chance rate 0.095; the measured random arm scores slightly above that at 0.1125, and prose comparisons in the text use the measured arm. *fmt* is the parseable-answer rate; *tgt ok* is the share of reply targets that were among the ten offered; *tgt hit* is the share that really received a reply that step; *jacc* is token Jaccard of generated text against the real next-step replies, for which the real-reply-versus-context reference is 0.029. **—** marks an axis that is not competed for our arms: a selector returns a selection rather than a document, so *fmt* and *tgt ok* would be 1.00 by construction; the minute is a uniform draw and not a learned factor (§3); the reply-target head the simulator deploys is the served-pool refit of §E, evaluated there on the event graph; and the own-count arm emits in only seven episodes, too few to carry a precision.

arm	fmt	ev/ep	count MAE	actor prec	lift	minute W_1	tgt ok	tgt hit	jacc
ours, matched count	—	2.00	1.25	**0.4958**	5.22	—	—	—	none
ours, oracle count	—	1.90	0.00	0.6028	6.35	—	—	—	none
ours, own count	—	0.06	1.84	—	—	—	—	—	none
Claude Opus 4.6	0.87	1.57	1.30	0.3662	3.85	1.73	0.40	0.113	0.021
Llama-4-Maverick-17B-128E	0.93	2.81	1.56	0.2618	2.76	1.01	0.47	0.149	0.022
GPT-5	1.00	5.47	3.73	0.2320	2.44	1.26	0.87	0.060	0.019
OSim (`cmu-lti/osim-8b`)	0.61	2.12	1.90	0.2226	2.34	1.96	0.44	0.063	0.025
Qwen2.5-14B-Instruct	1.00	3.00	1.48	0.2139	2.25	1.68	0.84	0.114	0.013
Llama-3.3-70B-Instruct	1.00	5.00	3.10	0.2100	2.21	1.55	0.55	0.081	0.021
Qwen2.5-7B-Instruct	1.00	3.00	1.48	0.1972	2.08	2.02	0.21	**0.174**	0.018
Qwen2.5-32B-Instruct	1.00	3.08	1.52	0.1844	1.94	1.03	0.87	0.106	0.014
Qwen2.5-3B-Instruct	0.42	0.92	1.68	0.1503	1.58	4.34	0.03	0.000	0.017
Llama-3.1-405B-Instruct	1.00	7.92	6.03	0.1444	1.52	1.13	0.66	0.067	0.025
gemma-3-4b-it	0.93	5.26	3.54	0.1405	1.48	2.06	0.54	0.060	**0.026**
Hermes-3-Llama-3.1-8B	0.97	3.11	1.62	0.1318	1.39	1.40	0.81	0.051	0.023
Phi-4-mini-instruct	1.00	1.99	1.26	0.1292	1.36	5.15	0.08	0.000	0.014
random over the same 20	1.00	2.00	1.25	0.1125	1.18	0.10	1.00	0.075	none
Gemini 3.1 Pro	0.96	0.00	1.90	—	—	—	—	—	none
OSim-Inst (`sunweiwei/OSim-Inst-8B`)	1.00	0.00	1.90	—	—	—	—	—	none

≈ 0.05, so exactly $n \sim \text{Uniform}\{0..4\}$ real actors are inserted. Over these 120 episodes the realised mean is 1.90 against a theoretical 2.0, putting the empirical chance rate at 0.095 rather than the pool's own 0.0024; every arm is scored against the realised figure. Precision here is therefore not comparable to §6, which pools candidate–step rows across every step under a different stratum weighting. And it is a single step, since none of the prompted models can be driven closed-loop for 96 steps at this prompt size. Our model was trained on this corpus; the others were not trained on this task.

K.2 Results

On *who*, the trained selector is in a different regime. At a matched count of two and with no oracle information it scores 0.4958 in Table S17, 1.35× the best prompted model, Claude Opus 4.6 at 0.3662, 4.4× the measured random arm. Arms answer different numbers of episodes, so the comparison is paired: episodes are resampled and both arms scored on the same draw. The margin over Claude is $+0.1146$ [0.0537, 0.1763] and over GPT-5 $+0.2638$ [0.2117, 0.3159] (Figure S5).

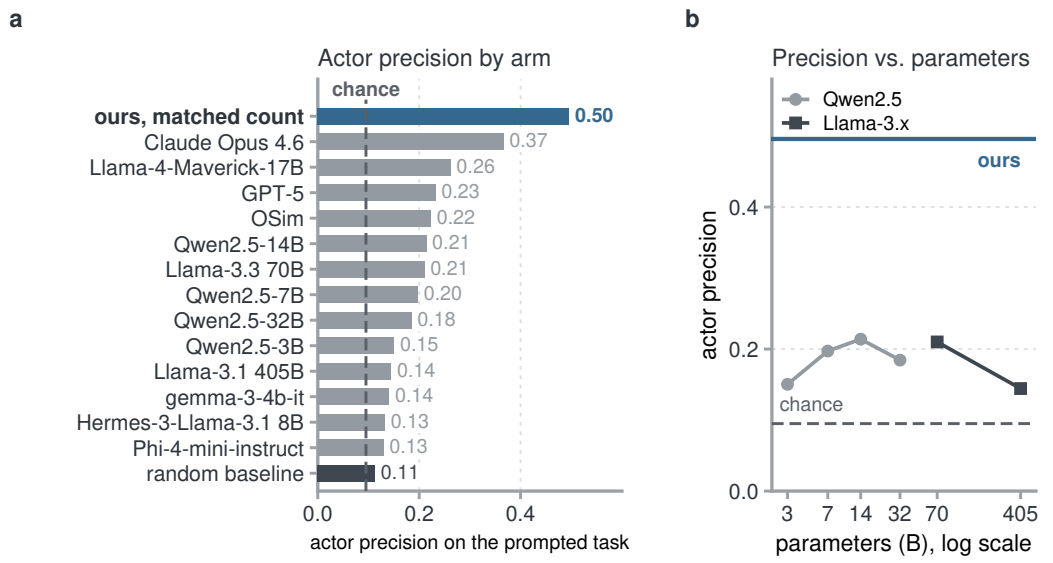

Figure S5: Actor precision on the identical single-step task, with chance dashed. Each episode plants 1.9 real actors among 20 candidates, so chance is 0.095. **(a)** every arm that names an actor. **(b)** two dense instruction families across a 135× parameter range against our matched-count score. Only the prompted arms were asked for text (Table S17).

It is not a model-size problem, and behavioural post-training alone does not supply it. Across Qwen2.5 at 3B to 32B and Llama at 70B and 405B, scores are flat over a 135× parameter range (Figure S5b). Actor precision divides by how many events an arm names, and the arms differ there (405B names 7.9 per episode against 32B's 3.1); at a matched count of two the ordering is unchanged: Llama-70B 0.329, Llama-405B 0.246. The three frontier systems spread wider than the parameter range does: Claude Opus 4.6 0.3662, GPT-5 0.2320, and Gemini 3.1 Pro predicts zero events on 115 of 120 episodes. OSim-8B, post-trained on behavioural data, reaches 0.2226 at a parse rate of 0.61.

A calibrated simulator declines to invent activity. Our model's own count over 20 candidates is 0.06 events per episode, because $\sum_u p_u \approx 0.05$ at the real base rate. The episodes plant forty times that, so it mostly declines to answer. It names an actor in seven of the 120, too few to carry a precision, which is why the matched-count row is the comparable one. Most prompted models emit one to eight events regardless; gemma-3-4b emits 5.26, a count MAE of 3.54.

Text. The arm we enter in this diagnostic is the selector alone, so it produces no text here; the text factor is exercised separately in §D. The prompted models produce 61–193 characters of topical prose, and the best of them reach 82–92% of the 0.029 lexical relatedness that a *real* reply has to the same context.

L Architectures, features and training settings

Every component of §3, with the settings it was fitted at. Nothing here is chosen per window or per seed.

Context encoder. Each factor that reads the world has its own copy of one shape: the 75-dimensional step-context vector through Linear$(75, 256)$ → GELU → Linear$(256, 256)$, giving $z \in \mathbb{R}^{256}$ at 85,248 parameters. The encoders are not shared between factors, in the deployed configuration or in the joint-objective one of §3.

The candidate features. The actor side reads 16 values per candidate–step. Eleven describe the candidate: `recency`, cumulative $\log(1 + n)$ event counts at 1 h, 6 h, 24 h and 7 d,

`enroll_cnt`, `inbound_recency`, `inbound_6h`, `n_communities_7d`, `community_affinity` and `account_age`; five more describe the step it sits in: hour-of-day sine and cosine, day-of-week, the global event count in the trailing hour, and the pool size. The hazard H and the deployed attention scorer read those 16 plus two more: the frozen actor score and the log gap since the candidate's last post. The reply-target head B reads ten values per (actor, candidate parent) pair: `node_age`, `replies_total`, `is_reply`, `author_events`, `prior_tie`, `self_reply`, `replies_1h`, `since_last_touch`, `same_community` and `thread_activity_1h`. The same builders compute these at training and at serving.

Actor factor. The hazard head is an MLP over the 18 features concatenated with z, trained by importance-weighted binary cross-entropy with negative subsampling: AdamW at 10^{-3}, weight decay 10^{-4}, one-cycle schedule with 5% warm-up, gradient-norm clip 1.0, batch 16,384 rows. The deployed scorer replaces that MLP with a feature tokeniser: each numeric feature is embedded through a 64-bin quantile piecewise-linear encoding and the resulting tokens attend to one another over 3 layers at $d_{\text{model}} = 96$ and feed-forward width 512 (368,545 parameters, fewer than the 404k MLP it replaces). It is fitted for 24,000 steps on 18,904,453 candidate–step rows, of which 13,700,737 come from real histories and the rest are harvested from the simulator's own drifted world (§C); the split is the first 80% of real steps for training and the last 20% for validation, so no step contributes to both.

Reply-target factor. B is a trunk $\text{Linear}(10 + 256, 512) \to \text{GELU} \to \text{Dropout}(0.1) \to \text{Linear}(512, 512) \to \text{GELU} \to \text{Dropout}(0.1)$ over the ten pair features and z_b, with a $\text{Linear}(512, 1)$ scoring head, 485,121 parameters including the encoder. It is trained by cross-entropy over 1,024 candidate parents sampled fresh per event: AdamW at 10^{-3}, weight decay 10^{-4}, one-cycle, batch 48 episodes, 6,000 steps, 50,000 training and 20,000 validation episodes, three seeds. The pool the episodes are drawn from is the seven-day one the simulator serves; the control of §E is the identical recipe on a six-hour pool.

Text factor. The base model is `cmu-lti/osim-4b` (Zhou et al., 2026), a Qwen3-4B derivative with 36 layers at hidden size 2560 and a 151,936-token vocabulary. Stage one adapts it on text alone with LoRA at rank 16, $\alpha = 32$, dropout 0.05, no bias term, on all seven projections (`q`, `k`, `v`, `o`, `gate`, `up`, `down`), giving 33.0M trainable parameters. Training uses AdamW at 10^{-4} with weight decay 0, one-cycle with 3% warm-up, gradient clip 1.0, bf16 autocast, batch 4 with gradient accumulation 4, 100,000 training examples for one pass, sequences capped at 1,024 tokens with the target allowed at most half of that. Stage two freezes that adapter and learns the world-state pathway alone: a $\text{Linear}(256, 8 \times 2560)$ producing eight soft-prompt tokens, 5,263,360 parameters, behind a tanh gate initialised at zero. It trains for 3,000 steps of AdamW at 10^{-3}, weight decay 10^{-4}, one-cycle, with examples packed to a 4,096-token budget per batch. Decoding is capped at 256 new tokens in the full-event run and 384 in the 116-event run of §D.

Rate model and multiplicity. G is a Poisson GLM on hour-of-day harmonics to order four and day-of-week indicators, fitted on the training segment and frozen, plus the single scalar level ℓ of Equation 2. The multiplicity table is a frozen 12×8 empirical distribution over (lifetime-rate bucket, event count) measured on the training segment. Neither reads anything the simulation emits.

Length factor. A linear quantile model at 101 levels over three inputs: the community, the length of the parent capped at 800 characters, and the length of the actor's own previous post capped at 400. Fitted on 99,994 training-segment replies, scored on 19,999 held out from them; the emitted reply is cut at the sentence boundary nearest the drawn target.

Windows. Each evaluation window is a 96-step, 24-hour closed-loop rollout. The starts come from a 13-point grid spaced evenly across the segment: §6 takes the even indices $2, 4, 6, 8, 10$ and §I.6 the odd $1, 3, 5, 7, 9, 11$. The level ℓ of Equation 2 is calibrated on the 28 days of real events immediately preceding each start.

Controller. $\tau = 0.4$; $\rho = 6$ logits over $K = 24$ steps; composition pinned to $m = (0.3165, 0.1701, 0.1888, 0.3246)$; attachment by Gumbel sampling at scale 1.0 from a 1,024-candidate pool redrawn every 32 actors. Three of these were chosen on validation, namely τ, (ρ, K)

and the resample rate, and **U** bounds the joint contribution of the first two together with composition pinning, which was not tuned.

The prompted arms. Every released model receives the identical rendering of the identical 120 episodes, under the instruction *"You simulate a social discussion platform. You will be shown the last few minutes of real activity, a fixed list of candidate users, and a fixed list of posts that could be replied to. Predict what happens in the next 15 minutes. Answer with JSON only."* followed by the schema

```
{"n_events": <integer 0-8>,
 "events": [{"cand":   <candidate number 1-20>,
             "minute": <integer 0-14>,
             "target": <reply-target nid, or null for a new top-level post>,
             "text":   "<what they write>"}]}
```

The answer is an index into the same twenty candidates our selector scores, not a free-form handle, so the choice being made is the same one on both sides; an answer counts as parseable if the array can be recovered from the response, which is the *fmt* column of Table S17. Decoding is greedy everywhere except GPT-5, which is run at its default temperature 1; a temperature-1 control on Llama-3.3-70B scores 0.2196 against its greedy 0.2100, inside the interval.

M Reproduction

The corpus is public (Baumgartner et al., 2020) and the runs reported here are on eight A100 80GB GPUs. The pipeline runs in six stages, each of which is specified above in enough detail to rebuild:

1. **Design facts, on the training segment.** The per-step rate series and the choice of G's functional form (§G); the loop-gain curve; the measured burstiness and recency composition (§F); the channels left out and what each costs (§2.2, §I.2).
2. **Component fitting.** G on the calendar; the multiplicity table; the hazard head; the level model; one round of dataset aggregation on harvested rollout features (§C).
3. **Actor scorer selection.** The architecture search of §I.5, 73 fits on a training-segment holdout, then a separate held-out pass on six windows disjoint from the five primary ones (§I.6).
4. **Reply-target closure.** The diagnosis of where generated candidates rank (§E), the serving-matched episode build, and the refit whose arms are compared on identical replayed pools.
5. **Text factor.** The text-only LoRA, then the frozen-LoRA zero-gated prefix stage and the length factor (§D), then the full-event run that produces the 1,575 complete events.
6. **Held-out evaluation.** Level calibration on the held-out segment, then the 75 rollouts of §6.1 (5 arms $\times$ 5 windows $\times$ 3 seeds) and the arm table and contrasts computed from them.