Grounded-Retrieval Adversarial Imitation Loop

Integrating Language, Agent, and World Models

Liv d'Aliberti
Princeton University
liv.daliberti@princeton.edu

Manoel Horta Ribeiro
Princeton University
manoel@cs.princeton.edu

Abstract

We present GRAIL, a grounded simulation framework that unifies Language, Agent, and World models. GRAIL retrieves candidate real next-actions from large behavior logs, allows an LM to reason and act (ReAct) to select among them, and includes a world-model component for counterfactual prediction. To support short-term alignment, a GAIL-style discriminator supplies an adversarial reward that aligns agent trajectories to human occupancy measures while mitigating the homogenization artifacts of "silicon survey" prompting. On a YouTube-related benchmark, GRAIL improves (i) trajectory fidelity to human behavior and maintains and analyzes (ii) downstream opinion-direction alongside classical baselines; full counterfactual stability assessment is left to future work. We release code, slates, and evaluation scripts [1] to catalyze grounded, agentic world-modeling research.

1 Introduction

Simulating human behavior with Language Models (LM) has coalesced around two dominant, complementary paradigms. The first, typified by Argyle et al. [1], treats a pre-trained LM as a stratified survey panel, and, by conditioning on rich demographic prompts, generates "silicon samples" whose aggregate response distributions closely track those of real-world sub-populations. The second are agentic-simulacrum approaches, pioneered by Park et al. [2]. Here, LMs are wrapped with external memory, reflection, and planning modules to create autonomous agents that pursue goals and interact coherently over extended time horizons [3].

Yet, both paradigms fall short of their full promise. Population-proxy methods treat each "silicon respondent" as a static and context-free bundle of demographic attributes, overlooking the evolution of behavior over time. Recent audits show they are prone to prompt-ordering biases [4] and fail to capture heterogeneity across sub-populations [5, 6]. Conversely, agentic-simulacrum systems can spin long, coherent narratives. Still, because they are spun from model priors, agents tend to drift [7], forget [8], or hallucinate [9] as simulations run, mistracking the very empirical diversity they aim to capture. Bridging these gaps requires grounding agent memories or goals in real behavioral trajectories while allowing demographic priors to update dynamically as the simulated life unfolds.

Here, we present **G**rounded-**R**etrieval **A**dversarial **I**mitation **L**oop (GRAIL): a hybrid simulation framework that is empirically grounded yet agentically expressive. Given an individual i at time t—say, we want to predict their stance on climate change $y_{i,t}$—we assume access to (i) static attributes x_i (demographics), (ii) a history of actions $a_{i,1:t-1}$ (e.g., news articles read), and (iii) past outcomes $y_{i,1:t-1}$. Our model comprises (see Figure 1):

An Environment Model (retrieval stage): retrieval-augmented generation fetches a slate of candidate next actions from large behavioral logs [10]. This builds upon past work in creating LM user-simulation pipelines grounded in real click-stream data [11, 12]. In our subsequent experiments, the Environment model is externally provided, but it could also be trained to maximize slate recall and tuned separately.

An Action Model (reason-and-act stage): an LM equipped with the ReAct loop reasons about the slate and selects one action [13], ensuring coherence with the user's prior trajectory. The action model is trained adversarially with a discriminator D_ω.

A Predictor Model (counterfactual stage): the synthetic trajectory $\langle a_{i,1:t-1}, a_{i,t} \rangle$ is fed into a causal predictor that produces calibrated counterfactual estimates of $y_{i,t}$ under alternative action sequences, leveraging the framework of Bottou et al. [14].

[1] https://github.com/liv-daliberti/grail-simulation/tree/main

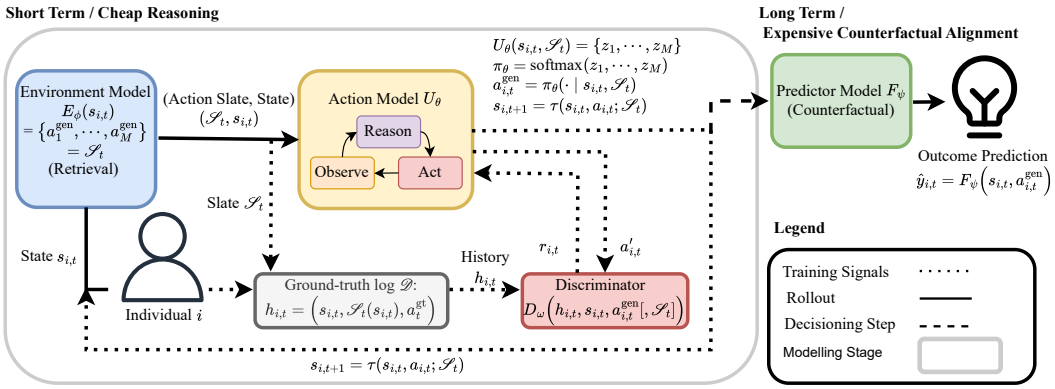

Figure 1: The environment model E_ϕ retrieves a slate $\mathcal{S}_t(s_{i,t}) = \{A_1, \ldots, A_M\}$ of candidate actions; the LM U_θ (ReAct) produces logits $z_{1:M}$, inducing $\pi_\theta = \text{softmax}(z)$, from which we sample a_t. The predictor F_ψ consumes $(s_{i,t}, a_{i,t})$ to estimate outcomes $\hat{y}_{i,t}$. Training-only (dotted): a ground-truth log $\mathcal{D}$ provides positives for a sequential discriminator $D_\omega(h_{t-1}, s_t, a_t[, \mathcal{S}_t])$, whose score yields reward r_t for PPO/GRPO updates to U_θ. The next step (if needed), $s_{i,t+1}$ can be calculated using deterministic function τ. Solid arrows denote rollout; dotted arrows denote training signals.

Many high-stakes decisions hinge on anticipating how human behavior evolves under policy or economic shocks [15–18], yet current LM simulations either mimic surveys without dynamics or spin agentic narratives that drift from reality [19–23]. GRAIL closes this gap, grounding agents in real behavior. Consider the use case of predicting users' opinions on contentious issues after watching a series of YouTube videos; as was done in an experiment by Liu et al. [24]. Here, GRAIL can predict how a user will change their opinion w.r.t. said issues by jointly modeling how individuals choose to navigate through YouTube videos and how they answer a survey (see Figure 2 for an example considering minimum wage).

Example User Prompt (Real Instance)

VIEWER 60-year-old, Caucasian/White (non-Hispanic) woman; conservative; earns $25,000–$29,999 per year; watches YouTube a few times a month.

Initial Viewpoint: Opposes a $15 minimum wage

CURRENTLY WATCHING *Briahna Joy Gray: DEBUNKING Corporate Myths About $15 Minimum Wage*

RECENTLY WATCHED (NEWEST LAST)

1. $15 minimum wage would cut 1.4 million jobs by 2025: CBO
2. Dems push forward with $15 minimum wage
3. Panel: Manchin Comes Out Swinging Against $15 Minimum Wage
4. Briahna Joy Gray: DEBUNKING Corporate Myths About $15 Minimum Wage

OPTIONS

1. Briahna Joy Gray: Progressives SHOULD WITHHOLD Votes For $15 Minimum Wage
2. The Cruelty of the $15 Minimum Wage
3. What the US Gets Wrong About Minimum Wage
4. THIS Is What Happened When Minimum Wage Was Hiked Over 20% to $14!

DECISIONS

Prediction by the Action Model: Which video will this user click on next?

Prediction by the Predictor Model: Will this user's opinion on a $15 minimum wage change after watching this sequence of content?

Figure 2: Example GRAIL use case: Learning from cheap, short-term reasoning to better gauge expensive, long-term decision-making.

2 The GRAIL Model

GRAIL trains the simulator policy π_θ with a sequential discriminator D_ω in an adversarial game that encourages on-support actions from the slate. Let $h_t = g_\omega(h_{t-1}, s_t, a_t)$ denote the discriminator's hidden state. We optimize

$$\min_\theta \max_\omega \underbrace{\mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}[\log D_\omega(h_{t-1}, s_t, a_t)]}_{\text{real human actions}} + \underbrace{\mathbb{E}_{\substack{s_t\sim\mathcal{D},\\ a_t\sim\pi_\theta(\cdot|s_t,S_t)}} \left[\log\big(1 - D_\omega(h_{t-1}, s_t, a_t)\big)\right]}_{\text{model simulated actions}}. \quad (1)$$

To convert the game into a dense learning signal for the policy, we define the per-step reward $r_t = -\log\big(1 - D_\omega(h_{t-1}, s_t, a_t)\big)$, and update π_θ over the slate with standard policy-gradient methods (e.g., GRPO) on $\{r_t\}$. Unlike cross-entropy, which relies upon prediction likeness to ground-truth label, this shaping performs occupancy-style matching: it rewards any choice the discriminator cannot reliably distinguish from human behavior and penalizes clearly off-support selections.

We update the slate policy $\pi_\theta(\cdot \mid s_t, S_t)$ using the discriminator-shaped reward r_t with a conservative GRPO/PPO-style objective. We compute advantages A_t (e.g., GAE with a learned value baseline) and optimize

a clipped surrogate so that large changes in the importance ratio $\rho_t = \pi_\theta(a_t^{\text{gen}} \mid s_t, S_t) / \pi_{\theta_{\text{old}}}(a_t^{\text{gen}} \mid s_t, S_t)$ do not induce destabilizing updates. When the logged action is present in the slate ($a_t^{\text{gt}} \in S_t$), we (optionally) add a light cross-entropy term $\lambda_{\text{ce}}\big[-\log \pi_\theta(a_t^{\text{gt}} \mid s_t, S_t)\big]$; this provides weak supervision without collapsing the policy to pure imitation, since the primary learning signal remains the discriminator-derived reward. In effect, the discriminator supplies dense feedback about human-likeness, the clipping and KL regularization keep updates on-support and stable, and the entropy term maintains diversity over plausible actions.

Our results rely on standard assumptions: Markovian state updates; high slate recall (the logged $a_t^{\text{gt}} \in S_t$ with high probability); and ignorability/consistency for counterfactual prediction. The retriever may be external and fixed; performance is reported conditional on its slates. We provide inference and training algorithms below.

Algorithm 1 (Rollout / inference). Given s_1, horizon T, and modules $(E_\phi, U_\theta, F_\psi)$, the retriever returns a slate $\mathcal{S}_t$, the LM policy U_θ produces logits $z_{1:M}$ and a slate policy π_θ, an action a_t is sampled, the predictor yields $\hat{y}_t$, and the state advances by $\hat{s}_{t+1} = f(s_t, a_t, \hat{y}_t)$.

Algorithm 1. Rollout / Inference (solid path)

Input: s_1, horizon T, models E_ϕ, U_θ, F_ψ
for $t = 1$ **to** T **do**
  $\mathcal{S}_t \leftarrow E_\phi(s_t)$; $z \leftarrow U_\theta(s_t, \mathcal{S}_t)$; $\pi \leftarrow \text{softmax}(z)$; $a_t \sim \pi(\cdot \mid s_t, \mathcal{S}_t)$; $s_{t+1} \leftarrow \tau(s_t, a_t, \mathcal{S}_t)$
end for
$\hat{y}_t \leftarrow F_\psi(s_t, a_t)$

Algorithm 2 (Training step). With minibatches from logs $\mathcal{D}$ and the current policy π_θ, update the discriminator, convert it to per-step reward $r_t = -\log(1 - D_\omega(\cdot))$, compute advantages A_t, and update the policy; optionally update F_ψ and (if used) E_ϕ.

Algorithm 2. Training step (one mini-batch)

Sample $(s_t, \mathcal{S}_t, a_t^{\text{gt}}) \sim \mathcal{D}$ and $(s_t, \mathcal{S}_t, a_t^{\text{model}}) \sim \pi_\theta$
Update D_ω with (1); compute $r_t = -\log(1 - D_\omega(\cdot))$ and advantages A_t
Update U_θ (PPO/GRPO style)
Optional: Update F_ψ on (s, a, y) (MSE or IPW) and train E_ϕ to increase slate recall

3 Experiments

Data. We evaluate on the naturalistic click–trace corpus of Liu et al. [24] that manipulates YouTube-style recommendation slant in two policy domains. Liu et al. gather both initial/final policy opinion and the intermediate sequence of videos offered to each individual, and the videos each person selected. For details about our pre-processing and data selection procedures, please see Appendix A.

Protocol. Each session is a finite-horizon MDP: state $s_{i,t}$ includes individual covariates and recent watch history; $\mathcal{S}_t$ is the recommended slate; y_t is a pre-treatment attitude index. Rollout follows the solid path in Fig. 1; dotted paths are training-only. We evaluate each cohort separately: Study 1 (Gun Control, MTurk), Study 2 (Minimum Wage, MTurk), and Study 3 (Minimum Wage, YouGov). All models receive the same prompt information; see Appendix B for prompt preparation.

Splits. We respect the original chronology and ensure no participant leakage across splits; within each study we use 90/10 train/validation where applicable and report per-study results.

Models. For detailed model setup, see Appendix C. We consider:

- KNN (TF–IDF k-NN): TF–IDF index on training split; score by sum of top-k=25 train hits; no adversarial alignment [25].
- XGBOOST (ranker): gradient-boosted pairwise ranker (rank:pairwise, NDCG@5) on the same TF–IDF features with early stopping; pick $\arg\max$ score [26].
- POPPROXY ("Silicon Sampling"): GPT-4O [27] prompted on the structured viewer snapshot + options; one-shot index prediction, no RL/discriminator.
- GRPO FINETUNING: Qwen2.5-1.5B [28] with memory/reflection but no retrieval or discriminator, finetuned with GRPO [29].
- GRAIL: Qwen2.5-1.5B [28] with only a sequential discriminator (GAIL shaping [30]) as reward, optimized via GRPO.[2]

[2] All baselines use the same prompt builder and slates.

Table 1: **Next-video accuracy on eligible slates by study (higher is better).** Validation accuracy is computed only on slates where the participant's true next watch appears among the candidates. Entries are means with 95% CIs via participant-cluster bootstrap (in brackets). Study 1 = Gun Control (MTurk); Studies 2–3 = Minimum Wage (MTurk/YouGov). **Bold** denotes the best method per column. The *Most-freq.* baseline always predicts the study-specific modal video; *Random-from-slate* samples uniformly. N (eligible slates): Study 1=548, Study 2=671, Study 3=1,200.

Method	Study 1 (Gun)	Study 2 (Wage)	Study 3 (Wage)
GRAIL	**96.4%** [94.4, 97.6]	**38.8%** [35.2, 42.6]	**48.5%** [45.7, 51.3]
GRPO	96.2% [94.2, 97.5]	36.0% [32.2, 39.4]	44.1% [41.2, 46.8]
XGBOOST	87.4% [84.4, 89.9]	32.9% [29.5, 36.6]	35.9% [33.3, 38.7]
KNN (TF–IDF/ST)	76.3% [72.5, 79.6]	35.5% [31.9, 39.2]	32.0% [29.4, 34.7]
GPT-4O	26.1% [22.6, 29.9]	28.6% [25.3, 32.1]	25.1% [22.7, 27.6]
Most-freq. baseline	54.0% [49.8, 58.1]	36.8% [33.2, 40.5]	45.0% [42.2, 47.8]
Random-from-slate	32.6% [28.9, 36.7]	25.6% [22.5, 29.1]	25.6% [23.2, 28.1]

3.1 Validating the Action Model

How does GRAIL compare with silicon sampling and off-the-shelf machine learning models in predicting users' next actions? Using data from Liu et al. [24], we consider the task of predicting which videos users are likely to click, taking into account users' watch history and demographic attributes.

Table 1 reports results from experiments considering users' choices when watching videos related to the minimum wage (where at each step, users had to choose among 4 videos). Across the Minimum Wage cohorts (Studies 2/3; four options per slate), discriminator-shaped LMs outperform classical rankers on eligible-only next-video prediction: GRAIL achieves 38.8% and 48.5% vs. XGBOOST's 32.9%/35.9% and KNN's 35.5%/32.0%, exceeding both the most-frequent baseline (36.8%/45.0%) and the random-from-slate rate (25.6%). On Gun (Study 1), the gap is larger still: GRAIL reaches 96.4% eligible accuracy, surpassing XGBOOST (87.4%) and KNN (76.3%), while "silicon sampling" with GPT-4O remains near chance (26.1%/28.6%/25.1% across Studies 1–3).

3.2 Is GRAIL Able to Predict Individual Change in Opinion?

Next, we evaluate GRAIL's performance on expensive, longer-term predictions. Specifically, using data from Liu et al. [24], we consider the task of predicting whether someone changed their opinion after watching a series of YouTube videos. We predict both the *direction* (lean anti / none / lean pro) and the *magnitude* (none / small / medium / large) of opinion change, given that users have watched a specific sequence of videos.

As shown in Table 2, classical ML remains strong: XGBOOST leads on all cohorts, with KNN close behind. GRAIL/GRPO match GPT-4O on the Minimum Wage studies (Studies 2/3) but lag on Gun (Study 1). All models beat the no-change baseline.

4 Conclusion

GRAIL outperforms classical rankers on short-term slate ranking across all three studies, including large gains on Gun (Study 1). For longer-horizon opinion direction, classical ML (XGBOOST/KNN) currently leads, while GRAIL/GRPO match GPT-4O on the Minimum Wage cohorts but lag on Gun. These results suggest discriminator-shaped RL narrows the LM–ranker gap on click prediction; future work should extend to additional datasets, refine the discriminator, and assess robustness/out-of-distribution.

Table 2: **Opinion-direction accuracy by study (higher is better).** Predicts the *direction* of post-study opinion change relative to pre-study (classes: lean anti / none / lean pro). Entries report validation accuracy with 95% CIs via Wilson (per-participant) (in brackets). Study 1 = Gun Control (MTurk); Studies 2–3 = Minimum Wage (MTurk/YouGov). **Bold** indicates the best method in each column. The *No-change baseline* always predicts "no change." N (participants): Study 1=162, Study 2=165, Study 3=257.

Method	Study 1 (Gun)	Study 2 (Wage)	Study 3 (Wage)
XGBOOST	**75.9%** [68.8, 81.9]	55.8% [48.1, 63.1]	**54.9%** [48.8, 60.8]
KNN	70.4% [62.9, 76.9]	**56.4%** [48.7, 63.7]	52.1% [46.0, 58.2]
GPT-4O	70.4% [62.9, 76.9]	54.5% [46.9, 62.0]	51.0% [44.9, 57.0]
GRAIL	56.2% [48.5, 63.6]	54.5% [46.9, 62.0]	51.0% [44.9, 57.0]
GRPO	56.2% [48.5, 63.6]	54.5% [46.9, 62.0]	51.0% [44.9, 57.0]
No-change baseline	7.4% [4.3, 12.5]	6.1% [3.3, 10.8]	5.8% [3.6, 9.4]

Broader Impacts

Grounded, slate-level simulators can help researchers audit recommender behavior and study how short-term exposure relates to downstream attitudes. At the same time, such systems could be misused for targeted persuasion or for building user-level profiles that amplify bias or polarization. We mitigate by using a previously published, de-identified dataset, releasing evaluation code and slates (not user-identifying data), and clearly documenting limitations and intended use. We encourage follow-ups to include fairness/robustness audits (e.g., stratified error reporting, position-bias checks) and abstention policies before any real-world deployment.

References

[1] Lauren P. Argyle, Ellen C. Busby, Nathaniel Fulda, Joshua Gubler, Charles Rytting, and David Wingate. Out of one, many: Using language models to simulate human samples. *Political Analysis*, 31(3):337–351, 2023. doi: 10.1017/pan.2023.2. URL https://doi.org/10.1017/pan.2023.2. Also available as arXiv:2302.12347.

[2] Joon Sung Park, Jeffery C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, 2023. doi: 10.1145/3544548.3581279. URL https://doi.org/10.1145/3544548.3581279.

[3] Guang Wang, Ye Xie, Yuzhe Jiang, Ajay Mandlekar, Chen Xiao, Yiming Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. arXiv preprint arXiv:2305.16291, 2023. URL https://arxiv.org/abs/2305.16291.

[4] Ricardo Dominguez-Olmedo, Moritz Hardt, and Celestine Mendler-Dünner. Questioning the survey responses of large language models. arXiv preprint arXiv:2306.07951, 2024. URL https://arxiv.org/abs/2306.07951.

[5] Angelina Wang, Jamie Morgenstern, and John P. Dickerson. Large language models should not replace human participants because they can misportray and flatten identity groups. arXiv preprint arXiv:2402.01908, 2024.

[6] Dai Li, Linzhuo Li, and Huilian Sophie Qiu. Chatgpt is not *a man* but *das man*: Representativeness and structural consistency of silicon samples generated by large language models. arXiv preprint arXiv:2507.02919, 2025. URL https://arxiv.org/abs/2507.02919.

[7] Eeman Majumder. Generative life agents: A novel framework for persistent, evolving personas with traceable personality drift. arXiv preprint arXiv:rs-7018899, 2025. URL https://doi.org/10.21203/rs.3.rs-7018899.

[8] Qian Zhao, Zhuo Sun, Bin Guo, and Zhiwen Yu. Memocue: Empowering llm-based agents for human memory recall via strategy-guided querying. arXiv preprint arXiv:2507.23633, 2025.

[9] Ahmed M. Darwish, Essam A. Rashed, and Ghada Khoriba. Mitigating llm hallucinations using a multi-agent framework. *Information*, 16(7):517, 2025. doi: 10.3390/info16070517.

[10] Patrick Lewis, Ethan Perez, Aleksandra Piktus, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS 2020*, 2020. doi: 10.48550/arXiv.2005.11401.

[11] Lei Wang, Jingsen Zhang, Hao Yang, et al. User behavior simulation with large language model based agents. arXiv preprint arXiv:2306.02552, 2024.

[12] Ruiyang Ren, Peng Qiu, Yingqi Qu, Jing Liu, et al. Bases: Large-scale web search user simulation with large language model based agents. In *Findings of EMNLP 2024*, pages 902–917, 2024.

[13] Shunyu Yao, Jeffrey Zhao, Dian Yu, et al. React: Synergizing reasoning and acting in language models. arXiv preprint arXiv:2210.03629, 2022.

[14] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, et al. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of Machine Learning Research*, 14:3207–3260, 2013.

[15] Robert E. Lucas. Econometric policy evaluation: A critique. *Carnegie-Rochester Conference Series on Public Policy*, 1:19–46, 1976.

[16] Emmanuel Saez, Joel Slemrod, and Seth H. Giertz. The elasticity of taxable income with respect to marginal tax rates: A critical review. *Journal of Economic Literature*, 50(1):3–50, 2012.

[17] Michael Woodford. *Interest and Prices: Foundations of a Theory of Monetary Policy*. Princeton University Press, 2003.

[18] Sebastian Funk, Marcel Salathé, and Vincent A. A. Jansen. Modelling the influence of human behaviour on the spread of infectious diseases: a review. *Journal of the Royal Society Interface*, 7(50):1247–1256, 2010.

[19] Ricardo Dominguez-Olmedo, Moritz Hardt, and Celestine Mendler-Dünner. Questioning the survey responses of large language models. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024. URL https://openreview.net/forum?id=Oo7dlLgqQX. NeurIPS 2024; OpenReview ID: Oo7dlLgqQX.

[20] Yuan Gao, Dokyun Lee, Gordon Burtch, and Sina Fazelpour. Take caution in using LLMs as human surrogates. *Proceedings of the National Academy of Sciences*, 122(24):e2501660122, 2025. doi: 10.1073/pnas.2501660122. URL https://www.pnas.org/doi/10.1073/pnas.2501660122.

[21] Maik Larooij and Petter Törnberg. Do large language models solve the problems of agent-based modeling? a critical review of generative social simulations, 2025. URL https://arxiv.org/abs/2504.03274.

[22] Simon Münker, Nils Schwager, and Achim Rettinger. Don't trust generative agents to mimic communication on social networks unless you benchmarked their empirical realism, 2025. URL https://arxiv.org/abs/2506.21974.

[23] Boming Xia, Qinghua Lu, Liming Zhu, Zhenchang Xing, Dehai Zhao, and Hao Zhang. Evaluation-driven development of LLM agents: A process model and reference architecture, 2025. URL https://arxiv.org/abs/2411.13768. v2 (2025-03-27).

[24] Naijia Liu, Xinlan Emily Hu, Yasemin Savas, Matthew A. Baum, Adam J. Berinsky, Allison J. B. Chaney, Christopher Lucas, Rei Mariman, Justin de Benedictis-Kessner, Andrew M. Guess, Dean Knox, and Brandon M. Stewart. Short-term exposure to filter-bubble recommendation systems has limited polarization effects: Naturalistic experiments on youtube. *Proceedings of the National Academy of Sciences*, 122(8):e2318127122, 2025. doi: 10.1073/pnas.2318127122. URL https://www.pnas.org/doi/abs/10.1073/pnas.2318127122.

[25] Bruno Trstenjak, Stjepan Mikac, and Davor Donko. Knn with tf-idf based framework for text categorization. In *Procedia Engineering*, volume 69, pages 1356–1364. Elsevier, 2014. doi: 10.1016/j.proeng.2014.03.129.

[26] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939785. URL http://doi.acm.org/10.1145/2939672.2939785.

[27] OpenAI. GPT-4o. Online, 2024. URL https://openai.com/index/hello-gpt-4o/.

[28] Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

[29] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

[30] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *ICML 2016*, 2016.

Table 3: **Unique content coverage (minimum-wage corpus).** Counts of unique items after preprocessing.

Split	Current videos	Gold videos	Candidate videos	Unique slates	Prompt texts
train	198	393	406	14,465	21,361
validation	158	252	299	2,068	2,403
overall	199	399	412	15,770	23,764

Table 4: **Unique participants by split (all issues).** Unique user IDs grouped by participant.

Split	Participants (all issues)
train	5,292
validation	587
overall	5,879

A Data Selection Procedures

Our data come from the naturalistic YouTube-style experiments of Liu et al. [24], who built a video platform that scraped real YouTube recommendations and then experimentally "slanted" them (e.g., 3/1 like–minded vs. 2/2 balanced) while allowing participants to choose up to five videos per session. The studies span two issues—gun control and the minimum wage—with pre/post attitude batteries and rich on-platform interaction logs (watch time, likes/dislikes, saves). We evaluate all three cohorts—Study 1 (Gun Control, MTurk) and Studies 2–3 (Minimum Wage, MTurk/YouGov). For corpus-level coverage, we summarize the **minimum-wage** arm—the primary source of our slates, choices, and outcome labels—reporting unique videos, candidate slates, and prompt texts in Table 3. For completeness, we also provide unique participant counts by split (Table 4) and by study (Table 5) across the full three-study dataset.

Starting from raw survey exports and platform logs, we link each participant's survey waves to their on-platform sessions using topic cues, embedded links, and nearest-in-time matching. We then reconstruct a step-by-step timeline for each session that identifies the video being watched, the set of candidate recommendations presented at that step, and the next video the participant chose. Because the platform's slate records can appear in several nested formats, we parse them robustly and normalize video identifiers to a consistent YouTube key; when titles are missing, we backfill them from external catalogs. We retain only *learnable* steps—those with a non-empty slate and an observed next click—and restrict the corpus to minimum-wage sessions using a combination of topic hints and the presence of wage-policy fields in the surveys. The resulting unique-content and slate counts are summarized in Table 3.

All model inputs are derived directly from these reconstructed steps. The text shown to models combines a compact profile summary (drawn from demographics and attitudes) with a numbered list of the recommended options for that step. For the next-video task, the learning objective is to pick the option the participant actually chose from the presented list. For opinion-change analyses, we use the study's pre- and post-survey measures to label both the *direction* of change (toward supporting, no change, toward opposing) and the *magnitude* of change (none, small, medium, large) based on the absolute shift. Where available, we summarize stance-coded exposure from rating logs and watch time to support rule-based and nearest-neighbor baselines.

We split the resulting interactions into a **90/10** train/validation partition, stratifying by survey wave and grouping by participant to prevent cross-person leakage between splits. Split-level coverage for content and slates appears in Table 3, while the number of unique participants per split (all issues) and their distribution across Study 1/2/3 are shown in Table 4 and Table 5, respectively. Although we keep all eligible steps, headline results emphasize the most demanding regime—slates with exactly four options. This preparation preserves the ecological validity of the original study (real recommendations and free choice) while yielding a clean, reproducible dataset for both slate ranking and per-person opinion-change prediction. We exemplify the MDP representation of our data in Figure 3.

B Prompt Preparation

We first filter to examples that are actually learnable: a step must show a non-empty recommendation slate and we must be able to identify the participant's next click among those options. For each

Table 5: **Participants by study (all issues).** Study 1 = Gun Control; Studies 2–3 = Minimum Wage.

Split	Study 1	Study 2	Study 3
train	1,355	1,479	2,458
validation	162	168	257
overall	1,517	1,647	2,715

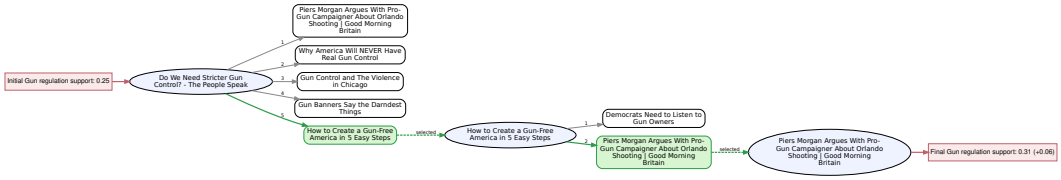

Figure 3: **Example session trajectory (Study 1: Gun Control).** Sequence of recommendation slates with the participant's chosen next watch (eligible-slate step). Illustrates the action-prediction setting in §3.

retained step, we build a two–message chat consisting of a concise instruction as the system message (see Fig. 4) and a structured user message with four blocks in fixed order: (i) a one–line viewer snapshot synthesized from demographics and early–wave policy cues, (ii) the current video (title, with optional identifier), (iii) a compact "recently watched" list (newest last, length–capped), and (iv) a numbered *OPTIONS* list derived from platform logs and normalized titles/IDs. The design exposes exactly what a human would use to choose the next video without leaking labels.

Next, we clean and normalize all text that enters the user message so formatting is consistent across sources. We standardize money/range notation, strip stray codes like "1) Conservative," remove cross–topic clutter (e.g., gun hints on a minimum–wage step), and backfill missing titles from external catalogs when needed. History is kept brief and legible, and the *OPTIONS* section is strictly numbered 1..N with names preferred and IDs as fallback. While Fig. 4 illustrates the fixed instruction shown to the model, the user message follows the standardized layout above on every example.

Finally, supervision and output constraints are embedded in the prompt format. As specified in Fig. 4, the instruction requires the model to think briefly inside `<think>...</think>` and then output *only* the chosen option's number inside `<answer>...</answer>`—no extra words, punctuation, or quotes—so evaluation reduces to exact index match. The gold label is the participant's real next click mapped to the numbered slate; we canonicalize by YouTube identifier and fall back to title matching when necessary. We cap prompt length (including a fixed history budget) for tractability and reproducibility, and store auxiliary fields (e.g., accuracy placeholders, mixture tags) alongside each example for training/analysis, though they never appear in the text shown to the model.

C Model Setup

All models operate on the same prepared minimum–wage split (90/10 train/validation) and consume an identical, standardized prompt (Fig. 4), with headline reporting on the 4–option regime. We compare a TF–IDF KNN and an XGBOOST pairwise ranker (classical, deterministic), a prompt–only GPT–4o baseline, and two Qwen 2.5–1.5B GRPO variants (accuracy–reward and discriminator–only); hyperparameters appear in the per–model tables, and evaluation uses Top–1/Top–2/Consistency with 95% CIs for Stage–A and accuracy/macro–F1/confusion for Stage–F, with per–example JSONLs released for audit.

Uncertainty and significance. For next-video (Stage-A), we report 95% confidence intervals via a participant-cluster bootstrap. For each study, we resample participants with replacement to the study's participant count, include all eligible slates for each sampled participant, and recompute accuracy. We use B=500 bootstrap replicates with a fixed seed (2024) and report percentile CIs. Concretely: Study 1 uses n_{groups}=162 participants and n_{rows}=548 eligible slates (analogously, Studies 2/3 use their per-study counts shown in Tables 1–2). For pairwise method comparisons, we form the bootstrap distribution of the accuracy *difference* (method A minus method B) and report its 95% percentile CI.

For opinion-direction (Stage-F), we report 95% Wilson intervals computed per participant (binomial), as reflected in Table 2. The same participant-cluster bootstrap (B=500, seed = 2024) can be used to obtain CIs for differences in opinion-direction accuracy when required.

```
You are choosing EXACTLY ONE item from a short slate for a specific
    viewer.

Input you will see:
  - Viewer profile and optional context/history
  - An "OPTIONS:" list with items numbered 1..N
    * Each item is shown as either a title (preferred) or an id

Your job:
  - Think briefly in <think>...</think> using the viewer's profile,
      context, and options.
  - Compare the top 2-3 candidates, then choose the single best
      option.
  - Never invent new items; choose only from the given OPTIONS list
      .

Output format (STRICT):
  - First output your hidden reasoning in <think>...</think>.
    * In your thinking, reference candidates by their numbers and
        names (or ids) to justify the choice.
  - Then output ONLY the chosen option's NUMBER inside <answer
      >...</answer>.
    * Do NOT output the name, id, or any extra  t e x t ONLY  the
        number.
    * Do NOT include punctuation, quotes, or a period after the
        number.

Examples of valid <answer>:
<answer>
3
</answer>

Examples of INVALID <answer> (never do these):
<answer>3.</answer>                  <- trailing period
<answer>"3"</answer>                 <- quoted
<answer>Option 3</answer>            <- extra words
<answer>Parkland ...</answer>        <- name instead of number
```

Figure 4: System prompt used for next-video selection.

C.1 KNN (TF–IDF)

We use a memory-based baseline because next-video choice on these slates is lexically grounded. We build structured prompt documents (shared builder) from the viewer/context text and candidate surfaces, and evaluate KNN over multiple feature spaces (TF–IDF, Word2Vec, Sentence-Transformer), reporting TF–IDF as the strongest in our setting. Configuration details (mirroring the sweep script) are summarized in Table 6. Scoring is *candidate-aware*: each option is queried against a training corpus and restricted to rows whose canonical title or video ID matches that option; scores aggregate the top-k similarities and we select the best k per study using the configured k-selection method.

How we build and "train" the model. No gradients: we fit a TF–IDF vectorizer (or train Word2Vec/encode ST) on the training split's prompt documents and cache label IDs/titles. At test time, we assemble a candidate-aware query (viewer/context + candidate surface + selection tokens), restrict to training rows whose canonicalized ID/title matches the candidate, sum the top-k similarities, and pick the highest-scoring option. We sweep k over $\{1, 2, 3, 4, 5, 10, 25, 50\}$ and select `best_k` per study using the `elbow` method (see Table 6).

Evaluation protocol (ranking and opinion change). For ranking (Stage–A) we report eligible-only accuracy with 95% CIs (participant bootstrap or Wilson), per-study breakdowns, and Most-Frequent/Random baselines. For opinion change, we use nearest-neighbor regression on participant-level documents to predict post-study opinion (and change), reporting MAE/RMSE/R^2 and direction accuracy. We also run `opinion_from_next`, which reuses the winning next-video configuration to tie exposure fidelity to opinion prediction.

Aspect	Setting
Representation	Structured prompt (viewer/context + candidate surfaces) via shared builder; default `max_history=12`
Feature spaces (swept)	`tfidf`, `word2vec`, `sentence_transformer`
Sentence-Transformer model	`sentence-transformers/all-mpnet-base-v2` (device: `cuda` if available)
k sweep	$k \in \{1, 2, 3, 4, 5, 10, 25, 50\}$
k-selection method	`elbow` (default; configurable via `--knn-k-select`)
Distance metrics	TF–IDF: `cosine`, ℓ_2; Word2Vec: `cosine`; ST: `cosine`
Text limits per space	`KNN_TFIDF_TEXT_LIMIT=5`, `KNN_WORD2VEC_TEXT_LIMIT=5`, `KNN_SENTENCE_TEXT_LIMIT=5`
Word2Vec training sweep	`size`$\in \{128, 256\}$; `window`$= 5$; `min_count`$\in \{1, 3\}$; `epochs`$=$ 10; `workers`$= 8$
Scoring	Candidate-aware query; sum of top-k similarities over training rows whose canonical ID/title matches the option
Uncertainty reporting	Eligible-only accuracy with 95% CIs (participant bootstrap or Wilson)

Table 6: KNN configuration aligned to the actual sweep grids (`training/training-knn.sh`).

Aspect	Setting
Objective / metrics	`multi:softprob` (classification); eval history logs `mlogloss` + `merror` when enabled
Text vectorizer grid	`tfidf`, `word2vec`, `sentence_transformer`
Learning rate grid	`eta`$\in \{0.03, 0.06\}$
Max depth grid	`max_depth`$\in \{3, 4\}$
Estimators grid	`n_estimators`$\in \{250, 350\}$
Subsample grid	`subsample`$\in \{0.8, 0.9\}$
Column sample grid	`colsample_bytree`$= 0.7$
Regularization grid	`reg_lambda`$= 0.7$; `reg_alpha`$= 0.1$
Tree method	GPU boosters if available; otherwise force `tree_method=hist` (auto-detected)
Inference over slates	Map global label probabilities to candidate IDs (fallback to canonicalized titles); select $\arg\max$ within slate
Uncertainty reporting	Eligible-only accuracy with 95% Wilson CIs; Most-Frequent and Random baselines

Table 7: XGBOOST slate configuration aligned to the actual sweep grids.

C.2 XGBOOST Ranker (multi-class)

We train a compact, deterministic multi-class classifier over the same prompt documents. Unlike a pairwise ranker, we use soft probabilities over the global label space and select among the slate by mapping candidate IDs/titles back to the model's label encoder (with a title-based fallback). This exploits strong lexical cues while learning shallow interactions; the full slate configuration appears in Table 7.

How we train and score the ranker. We build training documents and label IDs from the training split, fit the text vectorizer, encode labels, and train an `XGBClassifier` with the above grids (GPU boosters when supported; otherwise `tree_method=hist`). At test time we compute per-label probabilities for the prompt, map them to candidate IDs/titles, and choose the highest-probability option (see Table 7).

Evaluation protocol (ranking and opinion change). For ranking (Stage–A) we report eligible-only accuracy per study with 95% CIs and baseline comparisons, plus known-candidate coverage diagnostics. For opinion change (Stage–F) we train an `XGBRegressor` on participant-level documents to predict change (adding back the pre index) and include the pre-study index as a numeric feature; we report MAE/RMSE/R^2 and direction accuracy. We also run `opinion_from_next` using the winning next-video configuration to ensure exposure and opinion are evaluated under matched features.

Aspect	Experimental design
Inputs	Structured viewer/context snapshot plus a numbered list of slate options (identical builder to other baselines).
Training	None (zero-shot). Model is used as-is; no finetuning, retrieval, or discriminator.
Prompting & output	One-shot *index* prediction of the next watch; no chain-of-thought or tool calls.
Decode stabilization	Small, pragmatic sweep over sampling settings to reduce randomness; best setting chosen on validation and then fixed.
Opinion stage	Reuse the very same setup to classify direction of post-study opinion change at participant level.
Metrics	Next-video: eligible-only accuracy with 95% CIs. Opinion: direction accuracy with 95% CIs. Results in Tables 1 and 2.
Fairness	Same slates, same prompt text, same split protocol as KNN and XGBOOST; no extra context beyond what rankers see.

Table 8: **GPT-4O baseline: experimental setup.** A minimalist, zero-shot slate selector evaluated under the same inputs and splits as classical rankers. We stabilize decoding with a light sweep, pick the best setting on validation, and carry it forward unchanged to test and to the opinion task.

Aspect	Experimental setup
Base model / precision	`Qwen/Qwen2.5-1.5B-Instruct`, `bfloat16`, SDPA attention
Objective / reward	GRPO with `pure_accuracy_reward` (weight 1.0); 4 on–policy generations per prompt
Schedule	Cosine LR; lr $= 5 \times 10^{-5}$; warmup ratio 0.1; up to 10 epochs
Batching / memory	Per–device batch 8; grad accum 16; gradient checkpointing (non–reentrant)
KL control	Target 0.07; init coeff 0.2; horizon 50k; value loss coeff 0.25
Stability	Policy clip 0.05; value clip 0.5; max grad–norm 0.15; GAE 0.95; $\gamma = 0.99$
I/O caps	Prompt/completion caps $2024/512$ tokens; vLLM used for fast evaluation
Logging / checkpoints	Save every 10 steps (keep 500); W&B logging; push-to-hub enabled; seed 42
Prompting / decode	Same builder and index–only output as GPT baselines; greedy Top–1, small sample bundle for Consistency@4

Table 9: **GRPO (accuracy–reward) configuration.** We optimize the exact evaluation target (index accuracy) while constraining drift with a KL term. The input format and inference harness are the same as for the GPT baselines; only the policy weights differ.

C.3 GPT-4O Slate Baseline (POPPROXY)

We include a zero-shot, generative "silicon sampling" baseline using GPT-4O. At each eligible step the model is given the *same* structured viewer snapshot and enumerated candidate options as all other methods (Appendix B) and is asked to output the *index* of the next watch. There is no retrieval, no discriminator, and no reinforcement learning. To keep comparisons fair, we lightly sweep decoding settings (temperature/top-p/token cap) *only to the extent needed* to stabilize outputs, select the best setting by validation accuracy, and freeze it for test. The identical, frozen setup is then reused for the opinion-direction task at the participant level. Performance for both tasks appears in Tables 1 and 2.

C.4 GRPO: Qwen 2.5–1.5B

Our goal is to make the model select exactly one slate option by *index* under a strict output contract. A token–level LM loss is a poor proxy for this structured decision, whereas a *pure accuracy* reward scores the sampled answer against the gold index and directly optimizes the evaluation metric. We therefore fine–tune QWEN 2.5–1.5B-INSTRUCT with GRPO: on–policy sampling, value–guided updates, and a KL term that keeps the policy close to the instruction–following base model—preserving formatting fidelity while improving decision quality (Table 9). Inputs and runtime protocol (builder, slate text, index output, splits) are *identical* to the GPT baselines; we train and run inference through the same harness.

Aspect	Experimental setup
Base model / precision	`Qwen/Qwen2.5-1.5B-Instruct`; `bfloat16`; SDPA attention
Data / splits	`od2961/grail-gun`, `od2961/grail-wage`
columns	`state_text` (prompt), `next_video_id` (label)
Inputs / contract	Same builder as GPT and GRPO baselines; strict index-only output; token caps $2024/512$
Objective (only change)	GRPO with *discriminator–only* reward (GAIL-style); no explicit accuracy term; 4 on-policy generations / prompt
Schedule / batching / KL	*Identical to* Table 9 (cosine LR 5×10^{-5}, warmup 0.1, epochs up to 10, per-device batch 8, grad-accum 16, checkpointing; KL target 0.07)
Inference	Same harness as GPT/GRPO: greedy Top-1; small sample bundle for Consistency@4; vLLM for fast eval
Logging / artefacts	Save every 10 steps (keep 500); W&B logging; push-to-hub (`od2961/Qwen2.5-1.5B-OpenR1-GRAIL-{GUN,WAGE}`)

Table 10: **GRAIL (discriminator–only) configuration.** Matches Table 9 in every respect *except* the objective, which swaps the accuracy reward for a GAIL-style discriminator score computed on the sampled choice-in-context.

How we train and score. Starting from the instruction–tuned base, we present the same structured slate inputs used elsewhere and draw 4 on–policy completions per prompt; each completion is parsed for an index and scored by `pure_accuracy_reward`. Training uses cosine LR (5×10^{-5}) with warmup (0.1), up to 10 epochs, per–device batch 8, grad accumulation 16, and gradient checkpointing. KL control targets 0.07 (init 0.2, horizon 50k); we apply PPO–style clipping (0.05), value clip (0.5), max grad–norm (0.15), and GAE (λ=0.95, γ=0.99). Checkpoints are saved every 10 steps and logged to W&B. At inference, we reuse the *same* prompt builder and decoding protocol as the GPT baselines: greedy Top–1 for accuracy and a small sample bundle for Consistency@4.

Evaluation protocol. We follow the unified protocol. For ranking (Stage–A) we report eligible–only Top–1 accuracy with 95% CIs and Consistency@4, and release per–example JSONL artefacts (inputs, predictions, flags). For opinion change (Stage–F), we reuse the frozen GRPO model to predict direction at the participant level and report accuracy and confusion matrices alongside the JSONLs for reproducibility.

C.5 GRAIL: Qwen 2.5–1.5B (GRPO, with Discriminator Reward)

Rather than rewarding exact index match, GRAIL shapes behavior with an adversarial signal: a lightweight discriminator scores whether the sampled *choice-in-context* (viewer snapshot, condensed state, slate, and the chosen option surfaced by name) looks human–plausible. This preference–style signal nudges the LM toward in-support, contextually appropriate selections without using the gold index during training. Crucially, *everything else is held fixed relative to the GRPO–Accuracy run*: we use the same base model, data, prompt builder, output contract (index only), batching, scheduler, KL control, and inference harness as in Table 9. The only change is the *objective*: the accuracy reward is replaced by a discriminator–only (GAIL-style) reward. A concise summary appears in Table 10.

How we train and score. We fine-tune the same QWEN 2.5–1.5B base used by GRPO–Accuracy. For each prompt we sample 4 on-policy completions, parse the index from the strict output, and compute a discriminator plausibility score on the rendered choice-in-context; no gold index is used in the reward. Rewards are normalized and optimized with the same GRPO trainer and KL control as the accuracy run. Inference reuses the exact prompt builder and decoding protocol from the GPT/GRPO baselines (greedy Top-1 plus a small bundle for Consistency@4).

Evaluation protocol. We follow the unified protocol: for ranking (Stage–A) we report eligible-only Top-1 accuracy with 95% CIs and Consistency@4, releasing per-example JSONLs (inputs, predictions, flags). For opinion change (Stage–F) we reuse the frozen GRAIL model to predict direction at the participant level and report accuracy and confusion matrices, again emitting JSONLs for audit and replication.

Aspect	What we do in practice
When it's used	Enabled in the GRAIL runs to provide a plausibility reward for each sampled choice.
Device	Prefer a GPU when available, otherwise CPU; uses a fast tokenizer and robust handling of empty inputs.
Training vs. evaluation	Updated online during training; frozen during evaluation so rewards are computed without learning.
Classifier	Small Transformer text classifier with a two-way plausibility head; probability in $(0, 1)$ is the reward.
Input content	Viewer snapshot + condensed state + full slate (titles/IDs) + chosen option by name; no access to the gold index.
Reward scale	Probabilities are optionally rescaled/normalized and clipped by the trainer to keep magnitudes stable.
Stability	Small learning rate; gradient clipping; moderate input length (up to 512 tokens).
Mixture (optional)	A learnable mixture can combine environment and discriminator rewards.
Failure handling	If scoring or tokenization fails for a sample, its reward defaults to zero and training continues.

Table 11: **Discriminator setup at a glance.** Plain-language summary of how the plausibility classifier is trained and used during GRAIL. The training and inference harness match GRPO–Accuracy; only the objective differs.

C.6 Discriminator for GRAIL (implementation details)

Purpose and signal. The discriminator supplies a preference-style reward that scores whether a sampled *choice-in-context* (viewer snapshot, condensed state, the slate, and the model's chosen option surfaced by name) looks human–plausible. Given the rendered text x_t, a lightweight classifier returns a probability in $(0, 1)$; this value is used as the reward (optionally rescaled and normalized by the trainer). In the **reported GRAIL** runs we use a *discriminator-only* objective—no explicit index-accuracy term—while holding all other training and inference settings identical to the GRPO–Accuracy setup (Tables 9 and 10).

How the input is formed. The discriminator sees exactly what the policy sees at that step: (i) the viewer profile and recent history, (ii) the numbered slate with titles (or IDs when titles are missing), and (iii) the *chosen* option rendered by its surface name. The gold index/label is *never* included in the discriminator's input.

Training targets. At each eligible step we build positives and negatives: positives pair the context with the human's next choice; negatives reuse the same context with a non-chosen item when the model's pick differs from gold. Batches are drawn online from the current GRPO minibatch; if no pairs are available the update is skipped.

Model and training. We use a small Transformer text classifier (two-label plausibility head). Text is truncated to a moderate length (up to 512 tokens). The classifier is optimized with a standard cross-entropy objective, a small learning rate, and gradient clipping for stability. Inference returns the positive-class probability; if scoring fails for a sample, its reward falls back to zero and training proceeds.

Online updates and evaluation. During training, the discriminator is updated online from the current minibatch before its probabilities are used as rewards. During evaluation, the discriminator is frozen and used in inference mode so metrics reflect a fixed reward model.

Combining with other rewards. When both an environment reward and a discriminator reward are present, a simple learnable mixture can combine them; in this paper we run *discriminator-only* shaping to isolate its effect, so the mixture is not used.

License

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND. See the accompanying `LICENSE` and `NOTICE` files for details. Third-party datasets and model weights referenced in this paper remain subject to their respective licenses.

NeurIPS Paper Checklist

1. **Claims**
   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?
   Answer: [Yes]
   Justification: The abstract and introduction present GRAIL as a framework with three components (retriever, action model, predictor) and clearly state that our *empirical evaluation* focuses on next-video prediction and per-participant opinion-direction, with the counterfactual predictor described as part of the framework and deferred for future empirical study (§3; Tables 1 and 2). Claims about discriminator-shaped LMs improving short-term slate ranking and comparative opinion-direction results align with the reported experiments; broader predictor goals are explicitly scoped as future work.
2. **Limitations**
   Question: Does the paper discuss the limitations of the work performed by the authors?
   Answer: [Yes]
   Justification: The paper states assumptions and scope (fixed retriever, eligible-only slates, 4-option emphasis, minimum-wage corpus coverage, no participant leakage) in §3 and Appendix A, and notes standard modeling assumptions in the method section.
3. **Theory assumptions and proofs**
   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?
   Answer: [NA]
   Justification: The paper is empirical and does not present formal theorems or proofs beyond standard RL/GAIL objectives (Eq. 1).
4. **Experimental result reproducibility**
   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?
   Answer: [Yes]
   Justification: Data preparation and splits are specified (Appendix A); prompt format is given (Appendix B, Fig. 4); per-model configurations are summarized (Tables 6 to 10); and we state that per-example JSONLs and code/slates are released.
5. **Open access to data and code**
   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?
   Answer: [Yes]
   Justification: The paper links a public repository (footnote in the abstract) and names the cleaned datasets used for evaluation (Appendix C); instructions and per-example artifacts are described.
6. **Experimental setting/details**
   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?
   Answer: [Yes]
   Justification: We detail splits and hygiene (§3, Appendix A), prompt construction (Appendix B), and model/training settings in Tables 6 to 10, with additional discriminator implementation details in Appendix C.6.
7. **Experiment statistical significance**
   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?
   Answer: [Yes]
   Justification: Tables report validation means with 95% confidence intervals (Tables 1 and 2); CI computation is aligned with the setup text (participant-cluster bootstrap or Wilson, as described in the configuration sections).
8. **Experiments compute resources**
   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Compute resources section below specifies hardware and runtimes for each family of experiments: classical baselines (KNN, XGBOOST) on a 16-core CPU node with 64 GB RAM (KNN indexing/evaluation < 5 minutes per study; XGBOOST sweeps 10–30 minutes per study); the GPT-4O baseline via a hosted API (no local GPU usage; per-study evaluation in tens of minutes); and GRPO/GRAIL training on on a distributed cluster over NVIDIA A100 (80 GB) with gradient accumulation/checkpointing (reported checkpoints $\sim$6–10 GPU-hours per study; vLLM evaluation $\leq$30 minutes per study). Total project compute is within a few tens of GPU-hours and a few CPU-hours, with exact commands and sweep settings provided in the repository.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The paper uses a previously published, consented dataset [24], performs no new human data collection, and reports aggregate metrics without re-identification risk; code/data release follows standard reproducibility practices.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 4 ("Broader Impacts") outlines potential benefits (auditing recommenders; reproducible evaluation of exposure–attitude links) and risks (targeted persuasion, profiling, polarization), and details safeguards: use of a previously published, de-identified dataset; release of code/slates and checkpoints without hosted endpoints; clear licenses and model/data cards; and recommended fairness/robustness audits and abstention/monitoring policies prior to any deployment.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We release only de-identified, previously published data splits and trained checkpoints, each accompanied by model/data cards stating intended use (research/auditing), limitations, and potential misuse risks (e.g., targeted persuasion). No personally identifying information is released; IDs are canonicalized, and assets remain under their respective licenses and terms (project code under Apache-2.0; Appendix C.6). We provide evaluation scripts and static artifacts only—no deployment endpoints. Links:

    - https://huggingface.co/datasets/od2961/grail-wage
    - https://huggingface.co/datasets/od2961/grail-gun
    - https://huggingface.co/od2961/Qwen2.5-1.5B-OpenR1-GRPO-GUN
    - https://huggingface.co/od2961/Qwen2.5-1.5B-OpenR1-GRPO
    - https://huggingface.co/od2961/Qwen2.5-1.5B-OpenR1-GRAIL-GUN
    - https://huggingface.co/od2961/Qwen2.5-1.5B-OpenR1-GRAIL-WAGE
    - https://github.com/liv-daliberti/grail-simulation/tree/main/reports

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We credit all reused assets in-text and in the bibliography (e.g., dataset [24], models [27, 28]). The project code is released under Apache License, Version 2.0 (Appendix C.6); third-party datasets and model weights remain under their respective licenses/terms, which we reference where they are used and list in the repository `LICENSE/NOTICE`.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The paper states code/slates/evaluation scripts are released (abstract footnote), and the appendices document preparation, prompts, and evaluation; repository docs cover usage.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: No new human-subjects work or crowdsourcing was conducted by the authors; the paper reuses an existing published dataset.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether IRB approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No new data collection; prior work handled IRB.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

Answer: [Yes]

Justification: LLMs are central baselines and methods; usage is described in §3 and Appendix C (GPT-4o, GRPO, GRAIL), including prompt format (Fig. 4).

Compute resources. Classical baselines (KNN, XGBOOST) ran on a 16-core CPU node with 64 GB RAM; KNN indexing/evaluation completed in under 5 minutes per study and XGBOOST sweeps in 10–30 minutes per study. The GPT-4o baseline was evaluated via a hosted API (no local GPU usage); per-study evaluation completed within tens of minutes. GRPO and GRAIL were trained on a distributed cluster over NVIDIA A100 (80 GB) with gradient accumulation and checkpointing; the reported checkpoints required on the order of 6–10 GPU-hours per study, and vLLM-based evaluation finished within $\leq$30 minutes per study on the same machine. Total project compute (training+evaluation across studies) remained within a few tens of GPU-hours and a few CPU-hours; exact commands and sweep settings are included in the repository for reproducibility.