ArtificialGate

ArtificialGate Ltd. Research Manuscript

Kalman-Inspired Runtime Stability and Recovery in Hybrid Reasoning Systems

Barak Or
ArtificialGate Ltd., Israel
barak@artificialgate.ai

Abstract

Tool-augmented reasoning systems can depart from the intended task dynamics before final-answer failure is observable. When evidence is stale, delayed, or misrouted, a system may remain locally coherent while adapting to an incorrect evidence stream. We formulate this behavior as a runtime-stability problem rather than an output-accuracy problem. The contribution is not a Kalman filter for agents, but an operational runtime-stability vocabulary for hybrid reasoning systems: prediction, realized evidence, innovation, drift, recovery, and unrecovered failure are treated as measurable events in a finite-horizon reasoning trace. We define *cognitive drift* as the observable deviation that connects this vocabulary to runtime monitoring. In controlled multi-step HotpotQA tool-use episodes with persistent evidence mismatch, innovation monitoring detects drift in 79.2% of perturbed episodes. Recovery remains rare under sustained corruption even when detection is reliable, indicating that some runtime failures should be reported as unrecovered rather than treated as safe continuation.

Keywords. Hybrid Reasoning; Cognitive Drift; Runtime Vocabulary; Agent Monitoring; Innovation Drift; Runtime Recovery

1 Introduction

Modern reasoning systems increasingly combine learned representations with retrieval, non-parametric memory, tools, planning, and structured decision loops [1, 2, 3]. This development links earlier work on integrated planning [18] and neuro-symbolic reasoning [19, 20] with contemporary LLM agents that use explicit tools, web browsing, APIs, and long-horizon interaction [4, 5, 21, 22, 6, 7, 8]. Embodied and open-ended agents extend the same shift toward persistent interaction loops [23, 9]. Structured prompting and deliberation methods, including chain-of-thought [24], self-consistency [25], tree search [26], self-refinement [27], and reflection [28], have improved task-level performance. Evaluation, however, still often asks an output-centered question: did the model answer correctly, complete the task, or remain calibrated under corruption and distribution shift [10, 30, 11, 31, 32]?

For deployed agents, this question may be insufficiently timely. Behavioral testing has pushed beyond aggregate accuracy [33], agent benchmarks increasingly evaluate multi-turn behavior [12, 13], recent diagnostics have exposed tool-use failure modes [34], and AI-safety work has emphasized failures not captured by standard benchmarks [35, 36]. These evaluations rarely ask whether the reasoning process itself remains dynamically bounded when tools return stale, delayed, or semantically mismatched evidence. The gap addressed here is a runtime vocabulary: terms that let a system report when it is

This research was conducted under ArtificialGate Ltd. The author serves as Academic Director at the Google & Reichman Tech School, and as an External Lecturer at Technion – Israel Institute of Technology and Reichman University.

locally consistent, when it is drifting, when it has recovered, and when a detected failure remains unrecovered. An agent can become stable around an incorrect evidence source, reducing local inconsistency while leaving the intended task basin.

We study this phenomenon as *cognitive drift*: sustained deviation in the runtime dynamics of a hybrid reasoning system under evidence mismatch. The framework is Kalman-inspired only in a restricted operational sense: it borrows the separation between prediction, realized evidence, innovation, monitoring, and correction from classical filtering [37] and modern filtering theory [38]. It does not assume linear dynamics, Gaussian noise, known covariances, or Kalman-optimal updates for language-model reasoning states. The purpose of the analogy is to make runtime properties explicit: boundedness, detectability, recoverability, and unrecovered drift can be reported as distinct trace-level facts.

Contributions. This work makes three contributions. First, it formulates an operational runtime-stability vocabulary for hybrid reasoning systems, turning prediction, evidence mismatch, drift, recovery, and unrecovered failure into measurable trace-level objects. Second, it defines cognitive drift as a finite-horizon stability property that separates local innovation from semantic task preservation. Third, it instantiates the vocabulary in a lightweight monitor and evaluates it under controlled persistent evidence mismatch.

The remainder of the paper is organized as follows. Section 2 defines the runtime-stability vocabulary and controller. Section 3 describes the controlled HotpotQA tool-use experiment. Section 4 reports detection, recovery, and ablation results. Section 5 discusses implications and limitations; supplementary trajectories and technical details are provided in the appendix.

2 Operational Runtime Stability Vocabulary

Figure 1 summarizes the proposed vocabulary as a runtime loop. At each step, the system predicts the next state and expected evidence, compares that prediction with realized tool evidence, and monitors the resulting innovation. Recovery is activated only when windowed innovation drift crosses a calibrated threshold.

We model an agent through the observable quantities needed for runtime monitoring: an internal state representation $x_t \in \mathbb{R}^d$, observations $o_t \in \mathcal{O}$, and actions $a_t \in \mathcal{A}$. The state may be an embedding, memory summary, planner state, or other latent representation. A learned or implicit transition model predicts the next reasoning state,

$$\hat{x}_{t|t-1} = f_\theta(x_{t-1}, a_{t-1}, o_{t-1}). \tag{1}$$

An evidence map predicts the evidence expected under that state,

$$\hat{y}_{t|t-1} = h_\theta(\hat{x}_{t|t-1}, o_t). \tag{2}$$

At runtime, the system observes realized evidence $y_t \in \mathbb{R}^m$. We use the stochastic abstraction

$$x_t = f_\theta(x_{t-1}, a_{t-1}, o_{t-1}) + w_t, \qquad w_t \sim \mathcal{N}(0, Q), \tag{3}$$

$$y_t = h_\theta(x_t, o_t) + v_t, \qquad v_t \sim \mathcal{N}(0, R). \tag{4}$$

The covariance terms need not be known or correctly specified. Unlike in classical filtering [37, 38], they are used here only as calibration quantities for innovation normalization.

Definition 1 (Innovation)**.** The runtime innovation is the discrepancy between realized and predicted evidence:

$$\nu_t = y_t - \hat{y}_{t|t-1}. \tag{5}$$

For monitoring, we use scalar innovation energy

$$e_t = \frac{\nu_t^2}{S}, \tag{6}$$

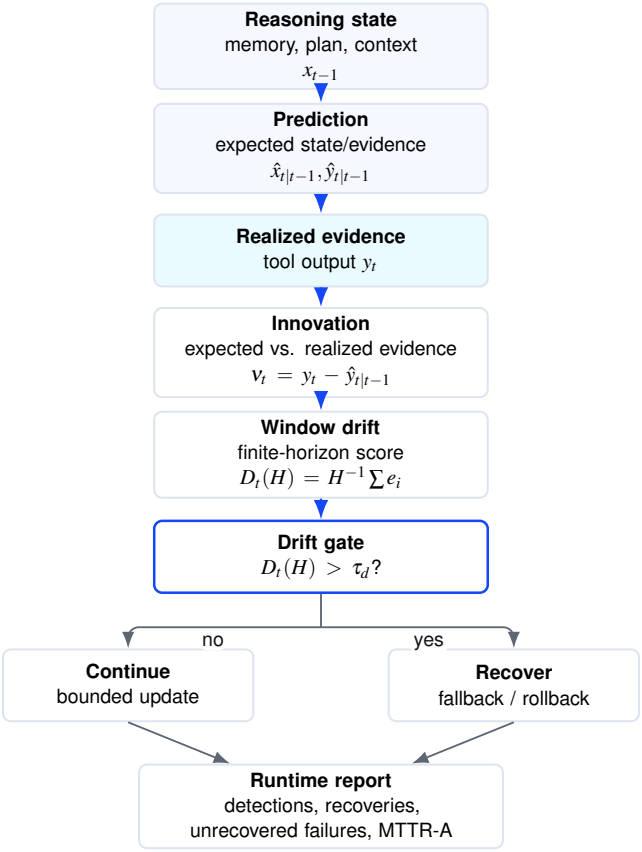


Figure 1. Operational runtime-stability vocabulary. The contribution is the naming and measurement of prediction, realized evidence, innovation, drift, recovery, and unrecovered failure inside a reasoning trace. The diagram is structural: it does not assume linear dynamics, Gaussian noise, known covariances, or Kalman-optimal control.

where S is estimated once from nominal trajectories:

$$S = \mathbb{E}_{\mathscr{R}_0}[(\nu_t - \mu_\nu)^2] + \varepsilon, \qquad \mu_\nu = \mathbb{E}_{\mathscr{R}_0}[\nu_t]. \tag{7}$$

We assume a nominal regime $\mathscr{R}_0$ in which

$$\mathbb{E}[e_t \mid \mathscr{R}_0] \leq \tau. \tag{8}$$

Definition 2 (Cognitive Drift)**.** For a window $H \in \mathbb{N}$, define

$$D_t(H) = \frac{1}{H} \sum_{k=0}^{H-1} e_{t-k}. \tag{9}$$

Cognitive drift occurs when $D_t(H)$ exceeds a calibrated threshold τ_d.

Innovation drift measures mismatch between predicted and realized evidence. It is not, by itself, a measure of task intent, a distinction also visible in behavioral evaluation, hallucination analysis, semantic-uncertainty estimation, and agent diagnostics [33, 14, 15, 34]. We therefore record semantic drift as a separate diagnostic quantity:

$$s_t = 1 - \cos(x_t, x_0), \tag{10}$$

where x_0 is the initial task representation.

Definition 3 (Runtime Stability)**.** A reasoning system is runtime stable if there exist $B > 0$, $B_d > 0$, and $\delta \in (0,1)$ such that, for all t,

$$\Pr(e_t \le B) \ge 1-\delta, \qquad \Pr(D_t(H) \le B_d) \ge 1-\delta. \tag{11}$$

Definition 4 (Recovery Time)**.** Let t_0 be the first detected drift time, $D_{t_0}(H) > \tau_d$. Recovery time is

$$t_r = \inf\{t \ge t_0 : D_t(H) \le \kappa\tau_d\}, \tag{12}$$

where $\kappa \in (0,1)$ is a bounded-stability factor.

We adopt Mean Time to Recovery for Agentic Systems (MTTR-A) [40]:

$$\text{MTTR-A} = \mathbb{E}[t_r - t_0], \tag{13}$$

where the expectation is over successfully recovered drift events. Unrecovered detections remain in the recovery-rate denominator, but are excluded from MTTR-A because t_r is undefined.

Formal consequences. The definitions above deliberately separate three runtime facts that are often conflated in output-only evaluation.

Proposition 1 (Bounded innovation implies bounded window drift)**.** If, for a fixed window ending at t, each innovation energy satisfies $e_{t-k} \le B$ for $k = 0, \dots, H-1$, then $D_t(H) \le B$.

Proposition 2 (Detection does not imply recovery)**.** Let t_0 be a detected drift time with $D_{t_0}(H) > \tau_d$. If $D_t(H) > \kappa\tau_d$ for all $t \ge t_0$, then no finite recovery time exists under Eq. 12.

Proposition 3 (Innovation stability does not guarantee semantic preservation)**.** Runtime stability as defined in Eq. 11 bounds innovation energy and drift score, but it does not imply semantic preservation, nor any bound $s_t \le c$ with $c < 2$.

Proofs are given in Appendix B. Proposition 2 motivates reporting unrecovered detections separately from MTTR-A, while Proposition 3 motivates tracking semantic drift as a diagnostic rather than treating low innovation as sufficient evidence of task preservation.

Algorithm 1 Runtime Stability Monitor and Recovery Controller

Input: window H, thresholds τ, τ_d, scale S, stability factor κ, recovery policy π_R
Output: drift alarms, recovery outcomes, and MTTR-A records

Initialize state x_0, window buffer $\mathscr{B} \leftarrow \emptyset$, mode nominal
for $t = 1, 2, \dots$ **do**
  Predict $(\hat{x}_{t|t-1}, \hat{y}_{t|t-1})$, observe tool evidence y_t, and compute $e_t = \nu_t^2 / S$
  Update $\mathscr{B}$ with e_t and keep only the most recent H energies
  If $|\mathscr{B}| < H$, continue monitoring; otherwise set $D_t(H) \leftarrow H^{-1} \sum_{e \in \mathscr{B}} e$
  Alarm: if $D_t(H) > \tau_d$ and mode is nominal, set $t_0 \leftarrow t$, apply π_R, and enter recovery
  Recovered: if $D_t(H) \le \kappa\tau_d$ and mode is recovery, record $t_r \leftarrow t$ and return to nominal
  Otherwise: apply the ordinary state update.
end for

The controller is intentionally lightweight. It instantiates the vocabulary rather than claiming optimal control. Tool fallback follows the broader tool-use paradigm in language agents [21, 22], while rollback and gain modulation act only on the runtime state and do not modify model parameters.

3 Experimental Setup

We evaluate the vocabulary on multi-step HotpotQA tool-use episodes in the distractor setting [39]. Each episode has horizon $T = 30$: the agent issues a retrieval query, receives a passage, and updates its latent reasoning state. The experiment is a controlled stability testbed rather than a task-optimized QA system, allowing runtime diagnosis to be studied separately from benchmark optimization [29, 33].

Predicted evidence $\hat{y}_{t|t-1}$ is instantiated as the predicted latent state. Retrieved passages are encoded with a pretrained sentence embedding model, and y_t denotes the retrieved-passage embedding. Thus,

$$\nu_t = 1 - \cos(\hat{y}_{t|t-1}, y_t), \tag{14}$$

with $e_t = \nu_t^2/S$ and $H = 3$.

Cognitive drift is induced by persistent tool misrouting. Starting at $t^\star = 5$, retrieval uses a real but incorrect query from another HotpotQA instance. The evidence remains fluent and task-like, but is systematically misaligned with the original intent. This setting tests semantic evidence corruption rather than random noise, complementing robustness, concept-drift, and adaptive-window monitoring settings [29, 16, 17].

The recovery policy instantiates π_R with three bounded actions: retry retrieval through tool fallback, reduce the state-update gain, and mix the current state toward the last pre-drift checkpoint using rollback weight β. Detection and drift thresholds are calibrated on nominal trajectories before perturbed episodes are evaluated.

Parameter	Value
Reasoning horizon T	30
Drift window H	3
Perturbation step $t^\star$	5
Episodes N	120
Innovation scale S	$\mathbb{E}[\nu_t^2]$ (nominal phase)
Detection threshold τ	90th percentile
Drift threshold τ_d	95th percentile
Stability factor κ	0.85
Nominal gain α	0.35
Rollback mixing β	0.20

Table 1. Experimental parameters and calibration settings. Thresholds and normalization are estimated from nominal trajectories.

4 Results

Runtime dynamics. Figure 2 reports aggregate behavior over $N = 120$ episodes. After $t^\star = 5$, the baseline often reduces innovation while semantic drift continues to grow, consistent with adaptation to an incorrect evidence stream. The recovery-aware agent maintains lower drift and lower semantic deviation.

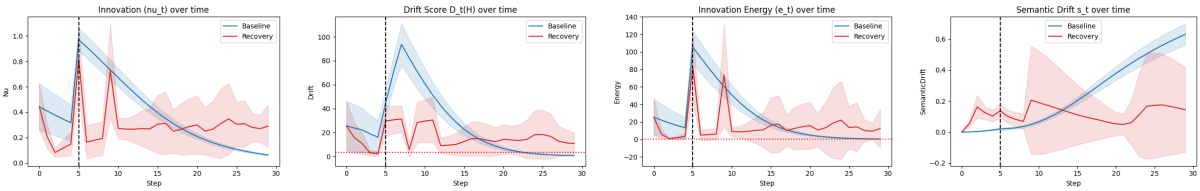
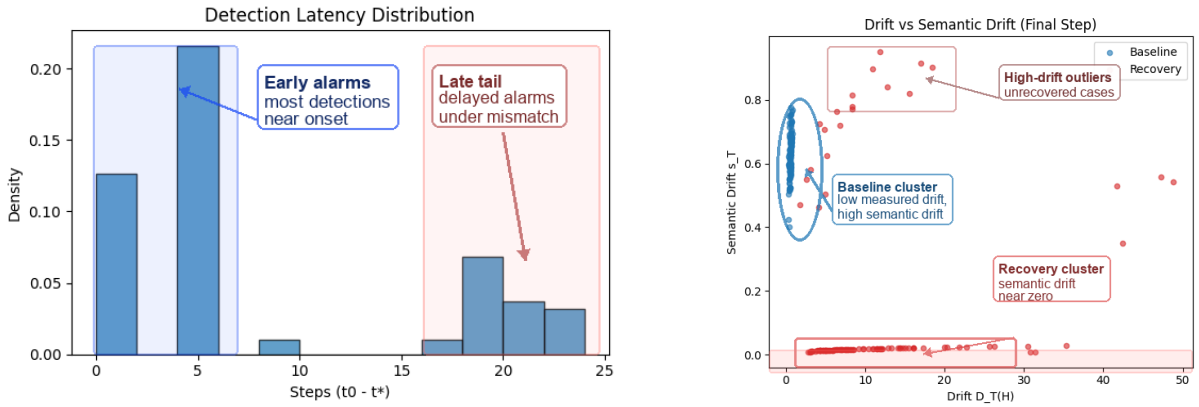


Figure 2. Aggregate runtime dynamics under persistent evidence mismatch. Blue denotes the baseline agent without recovery; red denotes the recovery-aware agent with the full controller. From left to right: innovation magnitude ν_t, innovation energy e_t, drift score $D_t(H)$, and semantic drift s_t. Shaded regions denote ± 1 standard deviation.

Detection and recovery. Cognitive drift is detected in 79.2% of perturbed episodes, with mean latency 7.81 steps after perturbation onset. Bounded stability is re-established in only 3% of detected events, while successful recovery has MTTR-A 4.00 ± 1.41 steps. Because MTTR-A is conditioned on recovered events, detection, recovery rate, and recovery latency should be interpreted jointly. Figure 3 separates timing from outcome. Alarms often occur near perturbation onset, but persistent mismatch creates a late tail. The scatter gives the central diagnostic: baseline runs can show low measured drift while accumulating high semantic drift, whereas most recovery-aware runs remain near zero semantic drift.



(a) Detection latency (b) Drift vs. semantic drift

Figure 3. Runtime instability diagnostics with interpretation. Blue denotes the baseline agent without recovery; red denotes the recovery-aware agent with the full controller. (a) Detection latency $t_0 - t^\star$: early alarms appear on the left; delayed alarms form the right tail. (b) Final drift score versus final semantic drift: the blue cluster marks local consistency with high semantic drift, the red lower band marks near-zero semantic drift under recovery, and high red points are unrecovered or weakly recovered cases.

Ablations. Table 2 isolates controller components. Removing tool fallback eliminates recovery despite high detection, suggesting that internal adaptation alone cannot repair a corrupted evidence channel. Rollback and gain modulation mainly affect the recovery-frequency/latency trade-off.

Condition	Det.	Rec.	MTTR-A
Baseline (no recovery)	0.07	0.00	–
Full controller	0.79	0.03	4.00 ± 1.41
No rollback	1.00	0.11	9.62 ± 7.75
No gain modulation	1.00	0.19	4.00 ± 0.00
No tool fallback	0.78	0.00	–
Delayed evidence perturbation	0.79	0.12	6.47 ± 4.01

Table 2. Ablation study of recovery mechanisms. Detection is the fraction of episodes in which cognitive drift is detected; recovery is the fraction of detected events that re-enter a bounded-stability regime. MTTR-A is computed only over recovered events.

5 Discussion and Limitations

The results support a modest but practically important claim: hybrid reasoning systems require runtime diagnostics in addition to final-answer metrics. Local consistency is ambiguous. A low innovation score can indicate reliable evidence alignment, but it can also indicate adaptation to unreliable evidence. Output-only evaluation distinguishes these cases late, if at all. Runtime reliability therefore benefits from process-level diagnostics that separate local consistency from semantic preservation, especially where calibration, uncertainty, agent, and risk evaluations already stress behavior beyond ordinary accuracy [10, 11, 31, 34, 36].

The study is intentionally limited. Prediction, innovation, and correction are monitoring primitives, not claims of linear state-space dynamics, Gaussian uncertainty, accurate covariance estimation, or

Kalman-optimal control. The experiments use controlled HotpotQA evidence-mismatch perturbations, and semantic drift is a diagnostic proxy. The contribution is operational: it makes one class of runtime misalignment nameable, measurable, and reportable before final-answer failure. It does not solve semantic alignment. The framework would be weakened if innovation drift routinely failed to precede semantic deviation, or if recovery statistics were insensitive to removing the proposed interventions.

6 Conclusion

This paper introduced cognitive drift as part of an operational runtime-stability vocabulary for hybrid reasoning systems. The central claim is that reliable agents should report not only whether an answer is correct, but also whether the reasoning trace remained bounded, drifted, recovered, or remained unrecovered along the way.

A Supplementary Runtime Evidence

The main text reports aggregate dynamics and final drift diagnostics. Figure 4 adds a representative single-episode trace showing the same failure mode at the trajectory level.

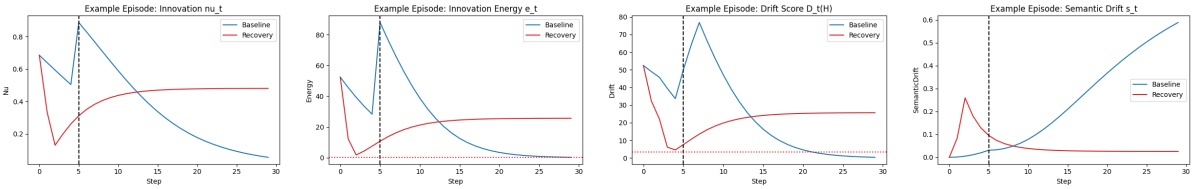


Figure 4. Representative single-episode trajectories. Blue denotes the baseline agent without recovery; red denotes the recovery-aware agent with the full controller. From left to right: innovation ν_t, innovation energy e_t, drift score $D_t(H)$, and semantic drift s_t. The dashed vertical line marks perturbation onset.

B Technical Notes

The framework is intentionally finite-horizon and diagnostic. The following short proofs clarify what is, and is not, implied by the definitions.

The proofs are placed in the appendix so that the main text can state the formal consequences without interrupting the presentation of the runtime vocabulary and empirical failure mode.

Proof of Proposition 1. Fix t and a window length $H \in \mathbb{N}$ with $H \geq 1$. By Eq. 9,

$$D_t(H) = \frac{1}{H} \sum_{k=0}^{H-1} e_{t-k}.$$

The assumption gives the pointwise inequalities $e_{t-k} \leq B$ for every $k \in \{0, \ldots, H-1\}$. Finite summation preserves order, hence

$$\sum_{k=0}^{H-1} e_{t-k} \leq \sum_{k=0}^{H-1} B = HB, \qquad D_t(H) = \frac{1}{H} \sum_{k=0}^{H-1} e_{t-k} \leq B,$$

where the final inequality follows because $H > 0$. The argument is deterministic and does not require independence, distributional assumptions, or Kalman-optimality. □

Proof of Proposition 2. Let $\mathscr{R}(t_0) = \{t \in \mathbb{N} : t \geq t_0 \text{ and } D_t(H) \leq \kappa\tau_d\}$. Equation 12 is equivalent to $t_r = \min \mathscr{R}(t_0)$ whenever this minimum exists. For any $t \in \mathbb{N}$ with $t \geq t_0$, the hypothesis gives $D_t(H) > \kappa\tau_d$, so $t \notin \mathscr{R}(t_0)$. Hence $\mathscr{R}(t_0) = \emptyset$, and $\min \mathscr{R}(t_0)$ does not exist in $\mathbb{N}$. Therefore no finite recovery time satisfies Eq. 12. Since $D_{t_0}(H) > \tau_d$ by assumption, the event is detected; since $\mathscr{R}(t_0) = \emptyset$, it is unrecovered. □

Proof of Proposition 3. We prove by counterexample. Fix $H \geq 1$, $S > 0$, and any admissible B, B_d, δ. Let $d = 2$, $m = 1$, $x_0 = (1,0)$, set $h_\theta(x,o) \equiv 0$, and set $y_t \equiv 0$. Then, for every state path, $\hat{y}_{t|t-1} = 0$, $\nu_t = 0$, $e_t = 0$, and $D_t(H) = 0$; hence Eq. 11 holds with probability one. Consider two admissible paths

with the same innovations: $x_t^{(a)} = x_0$ and $x_t^{(b)} = (\cos\phi, \sin\phi)$ for $t \geq 1$, where $\phi \in (0, \pi]$. Their monitored variables are identical, but $s_t^{(a)} = 0$ while $s_t^{(b)} = 1 - \cos\phi$. For any $c < 2$, choose ϕ close enough to π so that $s_t^{(b)} > c$. Thus no implication of the form Eq. 11 $\Rightarrow s_t \leq c$ for $c < 2$ is valid without an additional link between innovation and semantic state. Semantic drift must therefore be assumed, monitored, or controlled separately. □

References

[1] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. REALM: Retrieval-augmented language model pre-training. In *International Conference on Machine Learning*, 2020.

[2] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Empirical Methods in Natural Language Processing*, 2020.

[3] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuettler, Mike Lewis, Wen-tau Yih, Tim Rocktaschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, 2020.

[4] Ehud Karpas, Omri Abend, Yonatan Belinkov, Barak Lenz, Opher Lieber, Nir Ratner, Yoav Shoham, Hofit Bata, Yoav Levine, Kevin Leyton-Brown, Dor Muhlgay, Noam Rozen, Erez Schwartz, Gal Shachaf, Shai Shalev-Shwartz, Amnon Shashua, and Moshe Tenenholtz. MRKL systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning. *arXiv preprint arXiv:2205.00445*, 2022.

[5] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

[6] Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive APIs. *arXiv preprint arXiv:2305.15334*, 2023.

[7] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. *arXiv preprint arXiv:2307.16789*, 2023.

[8] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

[9] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A survey on large language model based autonomous agents. *arXiv preprint arXiv:2308.11432*, 2023.

[10] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017.

[11] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, 2017.

[12] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. *arXiv preprint arXiv:2308.03688*, 2023.

[13] Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. *arXiv preprint arXiv:2401.13178*, 2024.

[14] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Delong Chen, Wenliang Dai, Ho Shu Chan, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 2023.

[15] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*, 2023.

[16] Joao Gama, Indre Zliobaite, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Computing Surveys*, 46(4):1–37, 2014.

[17] Albert Bifet and Ricard Gavalda. Learning from time-changing data with adaptive windowing. In *SIAM International Conference on Data Mining*, 2007.

[18] Caelan Reed Garrett, Tomas Lozano-Perez, and Leslie Pack Kaelbling. Integrated task and motion planning. *Annual Review of Control, Robotics, and Autonomous Systems*, 2021.

[19] Tarek R. Besold, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, Kai-Uwe Kuehnberger, Luis C. Lamb, Priscila Machado Vieira Lima, Leo de Penning, Gadi Pinkas, and others. Neural-symbolic learning and reasoning: A survey and interpretation. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, IOS Press, 2021.

[20] Artur d'Avila Garcez and Luis C. Lamb. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *arXiv preprint arXiv:1905.06088*, 2019.

[21] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

[22] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023.

[23] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

[24] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

[25] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

[26] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, 2023.

[27] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and others. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023.

[28] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[29] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

[30] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, 2016.

[31] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, 2019.

[32] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

[33] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Association for Computational Linguistics*, 2020.

[34] Shuyan Zhou and others. Evaluating and diagnosing large language model agents. *arXiv preprint arXiv:2401.05459*, 2024.

[35] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mane. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.

[36] Toby Shevlane, Sebastian Farquhar, Ben Garfinkel, Mary Phuong, Jess Whittlestone, Jade Leung, Daniel Kokotajlo, Nahema Marchal, Markus Anderljung, Noam Kolt, and others. Model evaluation for extreme risks. *arXiv preprint arXiv:2305.15324*, 2023.

[37] Rudolph E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[38] Brian D. O. Anderson and John B. Moore. *Optimal Filtering*. Dover Publications, 2005.

[39] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[40] Barak Or. MTTR-A: Measuring cognitive recovery latency in agentic systems. Manuscript, 2025.