LoRe: 不確実性ゲートと KL 制約による LLM 方策事前分布の安全混合

LoRe: Uncertainty-Gated, KL-Constrained Mixing of LLM Policy Priors in DreamerV3

小笠真治
香川高等専門学校

概要

本稿は、モデルベース強化学習（MBRL）アルゴリズム DreamerV3 に対し、外部大規模言語モデル（LLM）の方策事前分布（policy prior）を必要なときだけ注入する軽量拡張 LoRe（Low-Regret LLM Prior）を提案する。LoRe の核は、(i) 状態依存の不確実性ゲートによる混合係数 $\beta(s)$、(ii) KL 逸脱のラグランジュ制御による安全側バイアス、(iii) API 予算・クールダウン・キャッシュによる実運用コスト抑制である。混合は単純なロジット加算

$$\ell_{\mathrm{mix}}(s) = \ell_{\mathrm{wm}}(s) + \beta(s)\ \mathrm{stopgrad}\big(\tilde{\ell}_{\mathrm{llm}}(s)\big), \quad \pi_{\mathrm{mix}} = \mathrm{softmax}(\ell_{\mathrm{mix}}) \tag{1}$$

で実装され、LLM 側へは勾配を流さない。MiniGrid-DoorKey-5×5 における早期スナップショット（約 10k-12.7k steps）では、10k 近傍の成功率（SR）が RL のみ ≈ 0.025 に対し LoRe ≈ 0.119 と立ち上がりが早い。終盤 12.7k 近傍でも LoRe は 0.094、RL のみは 0.021 と上回った（Seed=42, GPU=cuda）。過剰混合は不安定化を招くため、β と KL 目標の自動制御が有効である。DreamerV3 本体には改変を加えず、ロジット加算のみで統合できるため実装・計算コストは小さい [1, 2]。

1 序論

スパース報酬や手続き的依存をもつ環境では、初期探索の詰まりと行動の早期固定化が学習効率を阻害する。DreamerV3 は世界モデルによる想像訓練で多領域に通用するが、それでも初期段階の探索難は残ることがある。LLM は自然言語の手続き知識をもつため「鍵 → ドア → ゴール」のような高次の順序構造に関するヒントを提供し得るが、常時参照は過依存と汎化劣化のリスクを伴う。LoRe は LLM を常時の操縦者ではなく不確実時の“助言者”として扱い、必要時のみ prior を混合して探索の端緒を促す設計である [1, 2]。本稿の貢献は三点に集約される。第一に、不確実性駆動

の係数 $\beta(s)$ により状態依存に prior を混合する。第二に、混合による基底方策からの逸脱を KL 目標追従で安全側に抑制する。第三に、API 予算・クールダウン・キャッシュを備えた低侵襲の運用設計を示す。

2 関連研究

Dreamer 系と環境系. DreamerV3 は固定ハイパラで 150+ タスクを攻略する汎用 MBRL であり、本研究はその方策ヘッドに外部 prior を加える低侵襲拡張である。ベンチとして DoorKey (MiniGrid) はスパース報酬の古典的難題、Crafter/Craftax は探索・長期計画の能力を広く測る [3, 4]。

LLM× 行動/エージェント. LLM を行動計画・実行に活用する流れは、Zero-Shot Planners（言語 → 行動列）[5, 6]、ReAct（推論と行動の交互生成）[7]、Voyager (Minecraft での自己拡張的スキル獲得) [8]、Reflexion（言語による自己反省強化）[9] 等により進展してきた。本研究は LLM を直接の行動者にせず、既存 RL 方策の prior として混合する点が異なる。評価系としては AgentBench/SmartPlay が整備されつつある [11, 12]。

安全な混合・制約最適化. KL 制約に基づく方策更新は TRPO、確率的推論としての RL 理論等により位置付けられており、LoRe の KL 逸脱制御はこの系譜にある [13, 14]。また探索における不確実性は、世界モデルの分岐不一致 (disagreement) や価値分散、方策エントロピー等で測られてきた [15, 16]。

3 提案手法：LoRe

3.1 設計原則

LoRe は (a) 必要時のみ助言, (b) 基底方策からの逸脱抑制, (c) 実運用コスト抑制を同時に満たすことを目標とする。混合は加法ロジットで、DreamerV3 の学習則や損失の主構造に手を触れない。DreamerV3 実装や環境は既存リポジトリを参照する [17]。混合の中核は式(1)である。

3.2 LLM 事前分布の温度整合

LLM 側ロジットはスケールが異なるため、温度 $T_{\text{temp}} > 0$ を導入し

$$\tilde{\ell}_{\text{llm}} = \frac{\ell_{\text{llm}}}{T_{\text{temp}}}, \qquad T_{\text{temp}} = \arg\min_{t>0} \mathrm{KL}\left(\operatorname{softmax} \ell_{\text{wm}} \,\middle\|\, \operatorname{softmax} \frac{\ell_{\text{llm}}}{t} \right) \tag{2}$$

とする。これは知識蒸留や確率校正の考え方に近い［18, 19］。

3.3 不確実性ゲート $\beta(s)$

方策エントロピー $\hat{H}[\pi_{\text{wm}}]$、価値分散 $\widehat{\text{Var}}[V]$、世界モデル分岐不一致 $\widehat{\text{Dis}}$ を EMA 正規化し線形結合 $u(s)$ を作る。

$$\beta(s) = \beta_{\max} \cdot \sigma\big(\kappa(u(s) - \tau)\big) \tag{3}$$

高不確実な状態ほど prior を強める設計である［15, 16］。

3.4 KL 逸脱のラグランジュ制御

混合方策と基底方策の乖離 $\text{KL}(\pi_{\text{mix}} \,\|\, \pi_{\text{wm}})$ が目標 δ_{target} を超えた分だけペナルティを課す：

$$\mathcal{L}_{\text{KL}} = \lambda \,\text{ReLU}\big(\text{KL}(\pi_{\text{mix}} \,\|\, \pi_{\text{wm}}) - \delta_{\text{target}}\big), \tag{4}$$

$$\lambda \leftarrow \big[\lambda + \eta_\lambda \big(\text{KL}(\pi_{\text{mix}} \,\|\, \pi_{\text{wm}}) - \delta_{\text{target}}\big)\big]_+. \tag{5}$$

4 実験設定

環境: MiniGrid-DoorKey-5×5（鍵取得 → ドア解錠 → ゴール、スパース報酬）。**評価指標**: 成功率（SR）。**学習ステップ**: 約 10k-12.7k の早期遷移を観察。**比較**: RL のみ vs LoRe（β ゲート＋ KL 制約＋発火ポリシー有効）。DoorKey のタスク定義・難しさは公式ドキュメントを参照［2］。

5　結果

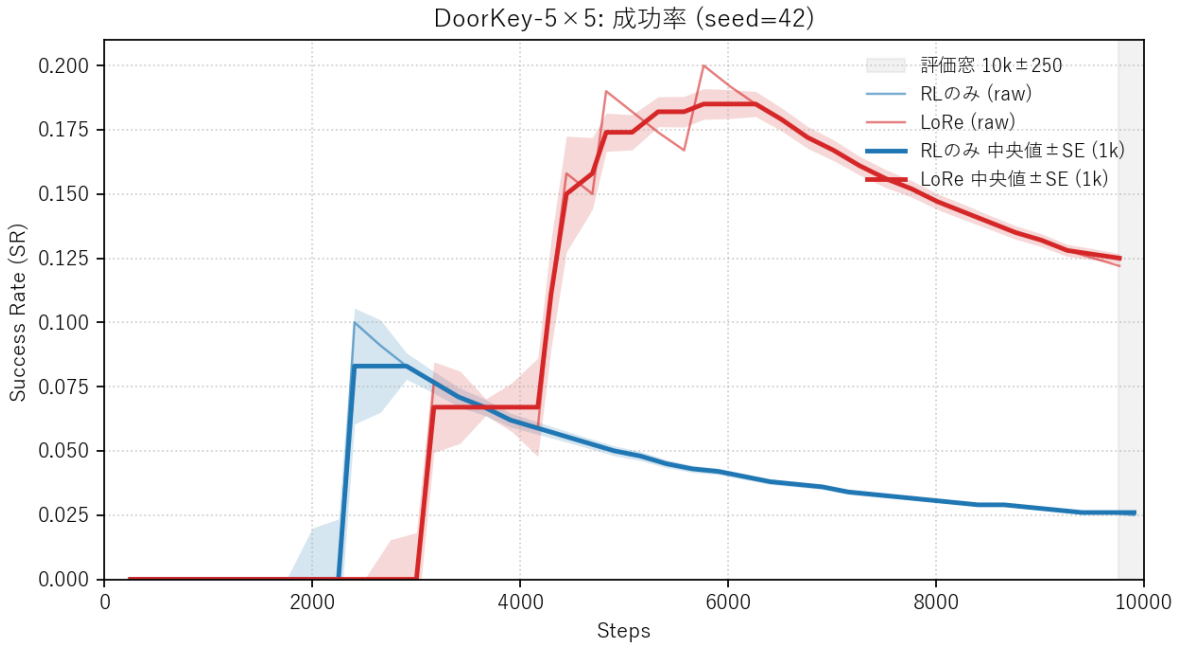

図 1: DoorKey-5×5 における成功率(中央値 ±SE；評価窓 = 各 1k steps；seed=42)。LoRe は 6-10k 域で一貫して RL のみを上回る。

図1は DoorKey-5×5（seed=42）における SR の推移である。10k±250 steps の窓で算出した中央値は、RL のみが 0.025、LoRe が 0.119 であり、差分は +0.094 であった。時点比較でも、6k では LoRe=0.20 / RL=0.045、8k では 0.15 / 0.03、10k では 0.12 / 0.025 と、LoRe は中盤以降で一貫して優位である。LLM API の総発火は 98 回で、中盤（6-8k）に集中した。これらは、β ゲート（式(3)）により探索のボトルネックで prior が選択的に強まり、同時に KL 逸脱制御（式(4), (5)）が過度な依存を抑制した結果と解釈できる。なお、本結果は単一 seed・早期スナップショットの観察であり、統計的検証には多 seed・長期学習が必要である。

6　考察

効き所とメカニズム. DoorKey のように明確な順序制約とスパース報酬がある課題では、LLM の手続き的知識は初期探索のボトルネックを緩和する。LoRe は不確実性が高い状態だけ prior を増やすため、LLM に全時刻を支配させず、DreamerV3 の学習ダイナミクス（想像訓練＋価値学習）を大きく乱さない。これは、ロボティクスで LLM の高次知識と低次価値推定をゲートで統合する手法（SayCan）の着想とも親和的である [21]。

制御の重要性. 温度整合（式(2)）がないと LLM ロジットは過鋭化し、方策が硬直

しやすい。単一スカラーの温度 T_{temp} でスケールを合わせるだけでも安定性が向上する。さらに、KL のターゲットを緩 → 厳へとスケジューリングすれば、初期の探索支援と後期の自律収束を両立できる［18, 19］。

位置付け. ReAct/Voyager/Reflexion 等の「LLM を主体的エージェントとする」路線と比べ、LoRe は RL 方策主体＋ LLM は advisor という安全側の設計であり、既存 MBRL の再利用性が高い。将来的には AgentBench/SmartPlay など言語記述を伴う環境での prior 混合の汎化検証も有用だろう［11, 12］。

7 アブレーション方針（今後の検証計画）

(1) 温度整合 T_{temp} の有無/適応更新、(2) KL 目標 δ_{target} のスケジューリング、(3) β の構成要素（エントロピー・価値分散・分岐不一致）の寄与、(4) API 発火ポリシー（クールダウン・キャッシュ・マクロ境界）の有無、(5) 発火頻度の早期学習への寄与、などを追試する。探索の不確実性指標については Plan2Explore/Disagreement 系の測度設計も併用して比較する［15, 16］。

8 限界と今後の課題

(i) 重み w_H, w_V, w_D はタスク依存であり自動調整（メタ学習/ベイズ最適化）が望ましい。(ii) KL 目標の自動スケジュール（学習進行度指標に基づく適応）を導入したい。(iii) LLM からサブゴール列や高次表現のみを prior として渡し、視覚のドメインギャップに強い抽象度整合を図る設計も検討に値する。(iv) 本稿の結果は単一 seed の早期スナップショット比較であり、多 seed・長期学習・環境多様化で統計的検証を拡張する必要がある［1］。

9 再現性メモ

リポジトリには､学習時に src[wm,llm,rand], prob_max, entropy_mix, KL(pi_mix||pi_wm) をログ出力する。API 呼び出しはキャッシュ命中率とクールダウンを併記。障害時は、世界モデル再構成誤差、価値の過信、β 飽和、KL 超過の恒常化をまず点検する。コードとディレクトリ構成は公開リポジトリを参照［20］。

再現性の要約表

環境 ID	MiniGrid-DoorKey-5x5-v0
乱数 seed	42
学習総ステップ（ログ）	20k steps
評価窓	各 1k steps（中央値, SE）
評価時刻	6k/8k/10k, 10k±250
LLM API 総発火（本ログ）	98 回

付録 A：実装要約

コード構成は agents/（DreamerV3: CNN+RSSM, actor-critic）, llm/（温度整合・予算・キャッシュ）, trainers/, utils/, main.py。ロギングは src[wm,llm,rand]、entropy_mix、KL(pi_mix||pi_wm) などを出す。DreamerV3 の実装方針や環境セットアップは既存資源に準拠 [17]。

付録 B：記号表

$\ell_{\mathrm{wm}}, \ell_{\mathrm{llm}}$	世界モデル/LLM のロジット
π_{mix}	混合方策
$\beta(s)$	不確実性ゲート係数（式(3)）
T_{temp}	温度整合の温度（式(2)）
δ_{target}	KL 目標値
λ	KL 罰のラグランジュ乗数（式(5)）
κ, τ	ゲートのスケール/しきい値

付録：学習ループ（高位擬似コード）

混合はロジット加算、β は不確実性ゲート、KL はラグランジュ制御。API は予算・クールダウン・キャッシュで節約。

```
for step in range(T):
  logits_wm = actor_head(s)
  if allow_call(s): # 予算・クールダウン・キャッシュ・マクロ境界
    logits_llm = call_llm(s)
    T_temp = estimate_temperature(buffer) # KL 最小化の1D探索
    u = uncertainty(s) # H[pi], Var[V], Disagreement の正規化結合
    beta = beta_max * sigmoid(k*(u - tau))
  else:
    logits_llm, beta = 0, 0
```

```
logits_mix = logits_wm + stopgrad(beta * logits_llm / max(T_temp, eps))
pi_mix = softmax(logits_mix)
a ~ pi_mix
s, r = env.step(a)

update_world_model()
kl = KL(pi_mix || softmax(logits_wm))
loss_actor += lambda * relu(kl - delta_target)
lambda = clip_pos(lambda + eta_lambda * (kl - delta_target))
```

参考文献

[1] Danijar Hafner et al. Mastering diverse domains through world models (dreamerv3). *arXiv preprint arXiv:2301.04104*, 2023. URL https://arxiv.org/abs/2301.04104.

[2] Farama Foundation. Minigrid - doorkey environment. https://minigrid.farama.org/environments/minigrid/DoorKeyEnv/, 2023.

[3] Danijar Hafner. Crafter: Benchmarking the spectrum of agent capabilities. *arXiv preprint arXiv:2109.06780*, 2021. URL https://arxiv.org/abs/2109.06780.

[4] Michael Matthews et al. Craftax: A lightning-fast benchmark for open-ended rl. *arXiv preprint arXiv:2402.16801*, 2024. URL https://arxiv.org/abs/2402.16801.

[5] Wenlong Huang et al. Language models as zero-shot planners. *arXiv preprint arXiv:2201.07207*, 2022. URL https://arxiv.org/abs/2201.07207.

[6] Wenlong Huang et al. Extracting actionable knowledge for embodied agents. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9058-9077, 2022. URL https://proceedings.mlr.press/v162/huang22a/huang22a.pdf.

[7] Shunyu Yao et al. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2023. URL https://arxiv.org/abs/2210.03629.

[8] Guanzhi Wang et al. Voyager: An open-ended embodied agent in minecraft. *arXiv preprint arXiv:2305.16291*, 2023. URL https://arxiv.org/abs/2305.16291.

[9] Noah Shinn et al. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023. URL https://arxiv.org/abs/2303.11366.

[10] Noah Shinn et al. Reflexion: language agents with verbal reinforcement learning. https://openreview.net/forum?id=vAElhFcKW6, 2023.

[11] Xiaogang Liu et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023. URL https://arxiv.org/abs/2308.03688.

[12] Yujia Wu et al. Smartplay: A benchmark for llms as intelligent agents. *arXiv preprint arXiv:2310.01557*, 2023. URL https://arxiv.org/abs/2310.01557.

[13] John Schulman et al. Trust region policy optimization. *arXiv preprint arXiv:1502.05477*, 2015. URL https://arxiv.org/abs/1502.05477.

[14] Sergey Levine. Reinforcement learning and control as probabilistic inference. *arXiv preprint arXiv:1805.00909*, 2018. URL https://arxiv.org/abs/1805.00909.

[15] Ramanan Sekar et al. Planning to explore via self-supervised world models. *arXiv preprint arXiv:2005.05960*, 2020. URL https://arxiv.org/abs/2005.05960.

[16] Deepak Pathak et al. Self-supervised exploration via disagreement. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 2019. URL https://proceedings.mlr.press/v97/pathak19a.html.

[17] Danijar Hafner et al. danijar/dreamerv3: Mastering diverse domains through world models. https://github.com/danijar/dreamerv3, 2023.

[18] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. URL https://arxiv.org/abs/1503.02531.

[19] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321-1330, 2017. URL https://arxiv.org/abs/1706.04599.

[20] Ogasa Shinji. kikuya1179/lore. https://github.com/kikuya1179/LoRe, 2025.

[21] Michael Ahn et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022. URL https://arxiv.org/abs/2204.01691.