RL^{2}EF:Improving Language Models using Environment Feedback via Reinforcement Learning in Healthcare Treatment Regimes

Anonymous Author
Anonymous Institution

Abstract

Large language models have shown promise in solving dynamic treatment regime (DTR) tasks. However, it is non-trivial to fine-tune the LLM for policy improvement while maintaining its general text understanding and generation ability. Inspired by the concept of Reinforcement Learning from Human Feedback(RLHF), we propose RL^{2}EF, an instructor-actor framework that utilizes environment feedback as natural preference data to fine-tune LLMs in the dynamic treatment regime settings. RL^{2}EF makes use of indications from the DTR environment for 'RL with Environment Feedback' (RLEF) fine-tuning to achieve best-of-both-world results on both the DTR and medical question-answering sides. We propose a bilevel-LLM architecture consisting of two language models - the policy model ActorLM to perform decision-making on the DTR environment and a knowledge representation model InstructLM to extract the environment's dynamic characteristics and summarize decision rules to assist ActorLM's decision-making. On the decision-making task, Experimental results show that the RL^{2}EF agent outperforms both existing RL-only and LLM-only policies on the *SimGlucoseEnv* treatment recommendation task by a considerable margin, enhancing sampling efficiency. On question-answering tasks, RL^{2}EF improves InstructLLM's question-answering ability on MMLU-Med and MedMCQA benchmarks.

Preliminary work. Under review by AISTATS 2025. Do not distribute.

1 INTRODUCTION

As the demand for effective treatment of complex diseases increases, personalized therapies tailored to patient-specific characteristics have become a critical topic in modern healthcare (Chan and Ginsburg, 2011). Traditional population-based treatment methods are shifting toward more individualized and dynamic approaches, particularly in the treatment of chronic diseases and multimorbidity, where therapeutic interventions can vary significantly between individuals. In this context, dynamic treatment regimes (Chakraborty and Murphy, 2014) have emerged as a critical component of personalized medicine. The DTRs are designed to develop sequential decision-making policies that adapt to changes in patient status and disease progression, providing opportunities for individualized and precise disease management.

Reinforcement learning (RL) has been effectively applied in DTR and showed promising potential. Zhu et al. (2020) applied RL algorithms to closed-loop blood glucose control in patients with type I diabetes, Raghu et al. (2017) investigated RL algorithm's effect on Sepsis treatment, and based on Zhu et al. (2020)'s work, Lim et al. (2021) further developed a modified RL algorithm that can take into account both safety and interpretability and apply it to blood sugar control. However, trained-from-scratch RL agents are not always reliable in DTR Luo et al. (2024a). RL remains limited for clinical deployment as it faces several challenges: (i) **sampling efficiency and safe exploration;** RL algorithms often require sufficient exploration and sometimes require potentially hazardous actions to learn from failure. This trial-and-error learning framework is unharmful in games, while unacceptable in clinical applications (Yu et al., 2021); (ii) **heneralization to unseen patients;** RL algorithms trained on a stationary environment are prone to the change of underlying Markov Decision Process (MDP), making it difficult to generalize to patients who have different Pharmacokinetic—Pharmacodynamic (PK/PD) dynamics (Luo et al., 2024b); (iii) **decision interpretability;** al-

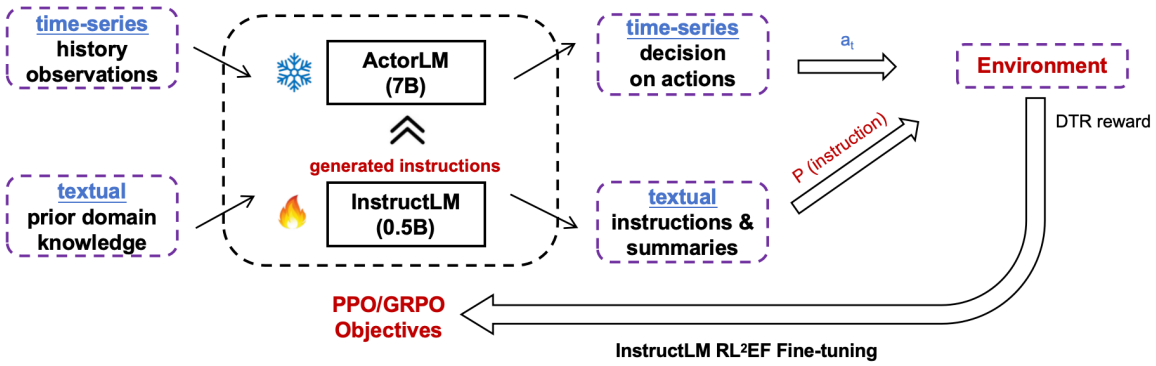

Figure 1: **The Framework of the LLM-RL^{2}EF Algorithm.** At every meta-timestep t (timestep in DTR environment), InstructLM summarizes the decision rules and generates prior-knowledge embedded instruction P by analyzing time-series history, while ActorLM uses both history and instructions to take action a_t. The DTR environment-generated reward was assigned to the last token in instruction P (combined with KL penalty in PPO) to be used for RL^{2}EF fine-tuning by optimizing PPO or GRPO objectives.

though there have been recent advances in improving the interpretability of RL (Puiutta and Veith, 2020; Glanois et al., 2024), interpreting AI-generated recommendations for clinicians with domain knowledge remains non-trivial, leading to hesitation in trusting and utilizing RL as a decision-support tool.

In recent years, large language models (LLMs) have shown significant promise in understanding and reasoning across general knowledge domains (Brown, 2020), including medical and healthcare contexts (Peng et al., 2023; Nazi and Peng, 2024). When it comes to applying LLMs to sequential decision tasks, frameworks such as ReAct (Yao et al., 2022), Reflexion (Shinn et al., 2024), and Retroformer (Yao et al., 2023) harness the in-context reasoning, reflection, and summarisation capabilities of LLMs to support decision-making. However, these models are typically pre-trained or fine-tuned and do not utilize data from agent-environment interactions. Instead, they rely solely on in-context learning, using prior knowledge and episodic memory to sustain general task performance. Despite their effectiveness, these approaches do not optimize the decision-making policy, representing, in our view, a missed opportunity for further improvement.

Optimizing LLMs' decision-making ability is challenging. LLMs are commonly trained by 2 means after pretraining: Supervised Fine-tuning (SFT) (Brown, 2020) and Reinforcement Learning with Human Feedback (RLHF)(Ouyang et al., 2022)(Shao et al., 2024; Chakraborty et al., 2024). SFT requires human-labeled question-answer(QA) pairs. In the DTR context, SFT needs multi-turn QA where each turn represents a step in the RL environment. RLHF demands expensive human preference annotations to train a reward model and then use policy optimization on the token level. Both means require expensive human-labeled data, which is rare in medical decision-making scenarios. Therefore, it becomes critical to find a more cost-effective way to finetune LLMs in DTR.

We position that environments can naturally provide reward signals that reflect preferences. We term this approach Reinforcement Learning with Environment Feedback (RLEF). In the medical field, clinical preference data is often costly and not readily available. This leads us to ask: **can we use environmental feedback in place of human feedback to fine-tune a language model, thereby improving both treatment performance (measured by RL rewards) and general medical question-answering ability?**

This paper aims to achieve two main objectives: (i) to explore the potential of LLMs in optimizing medical treatment regimes and (ii) to investigate how RLEF can enhance the medical QA ability of LLMs. We introduce RL^{2}EF, i.e. using RL to improve LLM via 'RL from environmental feedback', a framework that provides:

- **improvement on the RL side.** Weincorporate prior knowledge of LLM to address the limitations of RL in dynamic medical treatment regimes.
- **improvement on the RLEF side.** we use feedback from the treatment environment to enhance LLM's question-answering capability in the medical domain.

The RL^{2}EF framework uses an instructor-actor multi-LLM architecture consisting of an "actor" model (ActorLM) to make decisions and a smaller "instruct" model (InstructLM) to generate feedback or instructions to assist decision-making. We extend the conven-

tional policy optimization method, bridging the RLHF token-level MDP to the medical step-level MDP by RLEF. It is the first work in the healthcare community to investigate the potential of LLM-based policies in DTR and to achieve remarkable improvements in both treatment recommendation and medical QA metrics.

2 RELATED WORK

RL-based Dynamic Treatment Regime and Benchmarks RL applications in DTRs are divided into two main approaches: simulation-based and real-world data-based methods. Real-world data-based DTRs (Dann et al., 2014; Voloshin et al., 2019; Tang and Wiens, 2021) leverage observational healthcare data to train and evaluate RL models, but the lack of online testing in real-world data-based DTRs complicates the validation and iterative improvement of RL algorithms under real clinical conditions, often creating a gap between theoretical advances and actual clinical efficacy. In contrast, simulation-based DTRs (Zhu et al., 2020; Fox et al., 2020; Bhattarai et al., 2023) provide a controlled environment to test RL algorithms in various healthcare settings without ethical concerns of patient participation, enabling exhaustive testing in multiple hypothetical scenarios and allowing unlimited trial-and-error iterations in virtual patients, an approach that is impractical in real-world settings.

DTR-Bench (Luo et al., 2024b) established a unified framework to simulate various healthcare DTRs and compare the effectiveness of different RL algorithms in DTR applications including cancer chemotherapy, radiotherapy, glucose management in diabetes, and sepsis treatment. The research findings confirm the non-robustness and lack of adaptability of many RL algorithms in DTR applications, and underscore the necessity in the healthcare community to shed light on new perspectives for solving the pitfalls of pure RL algorithms.

RLHF for Human Preference Alignment Recent advancements in LLMs have significantly benefited from Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022), an approach that fine-tunes pre-trained models by incorporating human preferences into the training process. Based on the original Proximal Policy Optimization (PPO) RLHF (Ouyang et al., 2022), ReMax (Li et al., 2023) abandons the critic model and uses the optimal action under the current strategy as the baseline to reduce variance, achieving a lower computation and memory cost in fine-tuning than the conventional PPO approach. Group-relative Policy Optimization (GRPO) (Shao et al., 2024) inherits PPO RLHF's optimization objective, but substitutes the estimation of the critical model-related advantages with the group-relative Monte Carlo estimation, thus saving training costs and achieving better performance.

Despite these advances, there are challenges to scaling RLHF efficiently, especially when collecting massive amounts of high-quality human preference data and training a reward model requires great costs (Chaudhari et al., 2024). It is necessary and urgent to propose new methods for generating rewards more efficiently and effectively to guide language model fine-tuning.

Autonomous Language Model Agent for Decision Making Recent advancements in autonomous decision-making frameworks that use LLM have demonstrated significant potential in various tasks. These frameworks, which leverage the generalizability and knowledge-rich capabilities of LLMs, can be broadly categorized into open-loop and closed-loop approaches. Open-loop LLM-based frameworks, such as ReAct (Yao et al., 2022), Reflexion (Shinn et al., 2024), and ADaPT (Prasad et al., 2023), employ LLMs to generate "thoughts" of problem-solving based on observations without real-time feedback from the environment. However, these approaches often do not incorporate direct environmental rewards, limiting their adaptability. In contrast, closed-loop LLM-based frameworks, such as Refiner (Paul et al., 2023), Retroformer (Yao et al., 2023), and REX (Murthy et al., 2023), incorporate feedback mechanisms that facilitate iterative learning. These closed-loop frameworks acquire strong adaptability to the characteristics and dynamics of specific RL environments.

However, in these proposed frameworks, the foundation models used are either pre-trained or fine-tuned in advance before being applied to RL interactions. Therefore, the utilization of environmental feedback and rewards is only limited to the scope of prompt engineering, and the capabilities of the models do not get improved over the RL tasks. Some multi-modal LLMs like PaLM-E (Driess et al., 2023) are trained in RL tasks, such as robot control, for better capabilities in grounding languages to actions (Huang et al., 2022; Ahn et al., 2022) or decision makings. These frameworks bring about improvements to foundation models only in very limited task-related domains. However, bridging the gap between improving RL-related tasks and language-task performance for language models during RL interaction still needs further insight.

3 METHODOLOGY

Our approach incorporates two language models with distinct roles: a smaller, trainable LM (i.e., InstructLM) for generating textual representations from observations; and a larger, fixed LM(ActorLM) for making

treatment decisions based on the observation and generated representations. The objective is to utilize stepwise environmental rewards to improve the text generation capability of the InstructLM, therefore assisting the ActorLM to provide better treatment. In the section, We formulate the RL MDP in DTR, the convention RLHF MDP, and then introduce the extended MDP of RL^{2}EF, which incorporates environmental feedback into the existing RLHF framework to solve DTR tasks.

3.1 Reinforcement Learning for Dynamic Treatment Regime

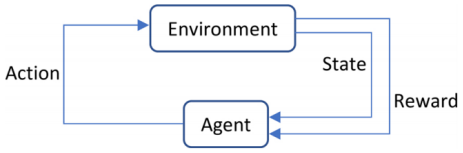

Figure 2: The Reinforcement Learning Framework.

A DTR Markov Decision Process is formally defined as a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$. The set $\mathcal{S}$ represents a finite set of states (i.e., clinical observation); $\mathcal{A}$ denotes a finite set of actions (i.e., drug dose). The state transition probability function $P_i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ characterizes the PK/PD dynamics of the patient i.

The primary objective in RL is to learn an optimal treatment policy $\pi^* : \mathcal{S} \to \mathcal{A}$ to maximize the expected cumulative discounted reward:

$$\pi^* = \arg\max_{\pi \in \Pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right], \qquad (1)$$

where Π is the space of all possible treatment policies, $\tau = (s_0, a_0, s_1, a_1, \ldots)$ represents a trajectory, and $s_0 \in \mathcal{S}$ is the initial state of the patient.

These fundamental concepts form the basis for various RL algorithms. In Appendix A, we provide an overview of several well-recognized RL algorithms in DTR, including Deep Q-Network (DQN) (Mnih, 2013) and Proximal Policy Optimization (PPO) (Schulman et al., 2017).

3.2 Reinforcement Learning from Human Feedback

Reinforcement Learning from Human Feedback uses RL to optimize human preference given a learned reward model and a base model (usually after SFT). Inspired by standard RL, RLHF uses a slightly different MDP for text generation. To differentiate from the MDP setup in dynamic treatment regimes, we use different notations to represent the token-level state and action in RLHF.

Consider a decoder-only language model π_θ with vocabulary $x \in \mathcal{V}$; the state at token n is defined as all tokens generated so far (i.e., $x_{1:n}$), and the action is the next possible token x_{n+1} . The policy is the next token prediction distribution of the language model $\pi_\theta^{\mathrm{H}}(x_{n+1}|x_{1:n})$. We expect the generated content to maximize the human preference oracle $r_H(x_{1:N})$, where N is the maximum sequence length, and x_N is the end-of-sequence token. Since the true human preference r_H is not directly accessible, we approximate it using a learned reward model R_ϕ parameterized by ϕ: $R_\phi(x_{1:N}) \approx r_H(x_{1:N})$. The primary objective is to find the optimal policy $\pi^{\mathrm{H}*}_\theta$ that maximizes the expected reward over sequences generated by the policy. Considering the outcome-only reward model(ORM), the optimization problem is formulated as:

$$\pi^*_\theta = \arg\max_{\pi_\theta} \mathbb{E}_{x_{1:N} \sim \pi_\theta} \left[\gamma^{N-1} R_\phi(x_{1:N}) \right]. \qquad (2)$$

Examples of RL algorithms used in RLHF are PPO (Ouyang et al., 2022) and GRPO (Shao et al., 2024). PPO introduces a clipping mechanism to control the magnitude of policy updates, leading to more stable training compared to the preceding actor-critic algorithms (Schulman et al., 2017; Ouyang et al., 2022). Based on PPO, GRPO uses the average rewards of multiple sampled outputs corresponding to the same question to estimate the advantage, greatly reducing the memory and computational costs of the value model. Detailed formulations of both algorithms can be found in Appendix B.

3.3 RL^{2}EF: Reinforcement Learning from Environmental Feedback via Reinforcement Learning

To bridge the gap between the MDPs used in standard DTRs and those in RLHF, we introduce RL^{2}EF. This approach extends RLHF to a multi-turn setting, allowing the language model to interact with the environment over multiple steps, receiving rewards at each step rather than only at the end of a sequence. By defining the MDP at the token level, we enable the application of standard reinforcement learning techniques to train language models in a sequential decision-making process.

The objective in RL^{2}EF is similar to RLHF. Instead of optimizing the single-turn response reward, we now optimize the expected return from the environment. Since all intermediate token rewards for a standard RLHF is 0, we can simplify the RL^{2}EF objective as

$$\pi_\theta^* = \arg\max \pi_\theta \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=1}^{T}\beta^t r_t\right], \tag{3}$$

where the environmental step $t = \sum_{i=1}^{t-1} N_i$ can be decomposed into token steps; $\tau = (s_1, a_1, s_2, a_2, \ldots, s_{N_T})$ is a trajectory sampled from the policy π_θ, and $\beta^t = \gamma^{N_t-1}$ is the modified discount factor in step level, and N_t is the number of tokens at step t. This formulation ensures that any LLM-powered MDP decision-making workflow can be trained directly via standard RLHF designs. Figure 1 shows the comparison between RL, RLHF and RL^{2}EF.

3.4 Instructor-Actor Framework for RL^{2}EF

We introduce the Instructor-Actor architecture, which contains two pre-trained LLMs for RL decision-making: **ActorLM** and **InstructLM**. This architecture was inspired by Reflexion (Shinn et al., 2024) and Retroformer (Yao et al., 2023) to incorporate a verbal feedback model that assists decision-making in the actor model. Figure3 shows the workflow of our architecture.

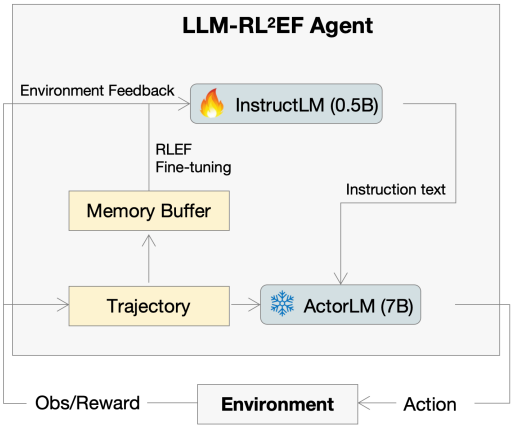

Figure 3: **Diagram of LLM-RL^{2}EF in DTR Environment.** The frozen ActorLM (labeled with a snowflake) acts as the direct policy to interact with the DTR environment, while the tunable InstructLM (labeled with a flame) leverages the trajectory buffer to RLEF fine-tune itself, aiming to generate better instructions in assisting of ActorLM's decision-making.

Notation and Formulation We denote the ActorLM as π_{θ_A} and InstrucLM as π_{θ_I}, parameterized by θ_A and θ_I respectively. Both ActorLM and InstructLM are functioning with specifically designed system prompts containing domain-specific prior knowledge and patient metadata, denoted as $\mathcal{P}_A$ and $\mathcal{P}_I$.

InstructLM is a trainable language model to extract specific dynamic characteristics of the MDP environment and summarize decision rules from the history of time series. As depicted in Figure 3, InstructLM provides expert prompts to instruct ActorLM toward better decision-making.

ActorLM is a parameter-frozen language model acting as a decision-maker with considerable capabilities in prior knowledge embedding and reasoning. As depicted in Figure 3, ActorLM is the direct policy to interact with the DTR environment.

Design considerations In practice, we can select a bigger frozen ActorLM model (e.g., 7B) with a smaller trainable InstructLM model (e.g., 0.5B). We do so for the following reasons: (i) large pre-trained LM already contains strong zero-shot reasoning capability, therefore setting a larger ActorLM is helpful for better decision making; (ii) fine-tuning on the large ActorLM is computationally expensive, so we freeze the ActorLM for inference only; (iii) fine-tuning the ActorLM directly might reduce ActorLM's general capability, causing "catastrophic forgetting" (French, 1999) or over-fitting on restricted DTR tasks. Thus, we try to improve ActorLM's performance not by fine-tuning ActorLM but by learning a better prompt given by the InstructLM.

For this consideration, we set the much smaller InstructLM LoRA-tunable (Hu et al., 2021), thus: (i) the model after optimizing for this specific RL task could generalize well on general medical text generation tasks; (ii) finetuning the much smaller InstructLM could avoid the high costs associated with expensive training. We provide the RL^{2}EF training paradigm using GRPO in Algorithm 1, and detailed designs of system prompts for ActorLM and InstructLM are listed in Appendix G.

Algorithm 1 RL^{2}EF via GRPO

Input: initial InstructLM policy $\pi_{\theta_{\text{init}}}$; instruction generating prompts $\mathcal{D}$; hyper-parameters ϵ, β, μ
Output: InstructLM policy π_θ
for iteration $= 1, \ldots,$ I **do**
  Sample a batch $\mathcal{D}_b$ from RL buffer $\mathcal{D}$
  Update the old InstructLM policy $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
  Sample G outputs $\{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot|q)$ for each prompt $q \in \mathcal{D}_b$
  Instruct ActorLM to interact with DTR to get environment reward $\{r_i\}_{i=1}^{G}$ for each sampled output o_i.
  Compute $\hat{A}_{i,t}$ for the t-th token of o_i through group relative advantage estimation
  for GRPO iteration $= 1, \ldots, \mu$ **do**
    Update the InstructLM policy π_θ by maximizing the GRPO objective Equation
  end for
end for

Table 1: **RL-side Test Results.** We mark the 1st highest returns on each patient cohort in red, and the 2nd highest in blue.

Patient Cohort	Naive baselines		RL-only baselines		LLM-only baselines		LLM-RL^{2}EF
	random	pulse	DQN	PPO	LLM	LLM w/ self-instruct	
adult - mean	201.01	130.48	275.62	244.45	207.94	247.62	254.84
adult - std	±100.25	±48.34	±1.68	±18.24	±31.16	±19.20	±27.52
adolescent - mean	96.97	131.11	217.09	202.55	149.96	181.69	197.01
adolescent - std	±64.77	±30.19	±61.81	±77.02	±48.48	±55.26	±66.24
training steps	-	-	288000	288000	-	-	5760

4 EXPERIMENTAL RESULTS

Here we introduce our experiment setup and present empirical results on both RL and RLEF tasks using the RL^{2}EF training framework.

4.1 DTR Environment - *SimGlucoseEnv*

We investigated *SimGlucoseEnv* - a simulation-based insulin administration environment for Type-1 diabetic patients. In *SimGlucoseEnv*, blood glucose dynamics are determined based on real-world data from 300 patients, covering a variety of metabolic parameters and demographic characteristics. The dynamics are formulated in ordinary differential equations (ODEs), which are developed based on computational models that simulate interactions between insulin dosing, carbohydrate intake, and glucose metabolism. *SimGlucoseEnv* depicts how glucose and insulin levels in the bloodstream are influenced by various processes such as glucose absorption, renal excretion, insulin fluxes, and insulin degradation (Man et al., 2014). The rationale behind choosing *SimGlucoseEnv* is: (i) as a simulation environment, it enables us to conveniently conduct large amounts of training and testing on it with different settings without concerns on data volume or balance; (ii) it has a limited number of variables, allowing us to examine the model's behavior in a more controlled setting.

In this work, we follow the environment setting of Luo et al. (2024b): the environment is updated at 5-minute intervals, and termination occurs if the basal plasma glucose level falls below 10 or exceeds 600. If neither condition is met, the environment continues for 24 hours (i.e., 288 steps). We set the environmental reward based on risk indices that encourage the agent to take action to reduce the risks related to hyperglycemia or hypoglycemia. The formulation of the risk function along with the entire *SimGlucoseEnv* ODE formulation and descriptions of each true environment state are detailed in Appendix C.

4.2 Experiment Setup

Treatment Policy Evaluation on DTR Environment (RL-side) For the *Simglucose* DTR environment, we set the following training mode: each policy was only trained on one adult patient, and then underwent the final test on 8 patients (four adults and four adolescents).

Baseline policies are divided into three categories: **naive policies**, **RL-only policies** and **LLM-only policies**. Naive policies included "random-0.1" and "pulse-0.05" (defined in Appendix D), RL-only policies included DQN and PPO, while LLM-only policies contained LLM and LLM w/ self-instruct, which calls itself (the larger pretrained ActorLM) to instruct decision-making. The LLM model used was Qwen2-7B (Yang et al., 2024). For RL policies including DQN, PPO, and LLM-RL^{2}EF, we trained 288k steps each and then underwent the final test on 8 patients each for 20 episodes. For learning-free policies, including 2 naive policies and 2 LLM-only policies, we directly underwent the final test without any training. The hyper-parameter configurations for all policies are listed in Table 5 in Appendix D. We collected each algorithm's best performance over all hyper-parameter configurations.

Table 2: Medical Knowledge Evaluation Datasets

Benchmark	**Question Number**	**Glucose-related Question Number**
MMLU-Med	272	63
MedMCQA	4183	144
Total	4455	207

Language Task Benchmarking on Medical QA We evaluated InstructLM's linguistic performance under the RL^{2}EF training framework on general and diabetes-related medical tasks by selecting two main LLM medical knowledge evaluation benchmarks: MMLU-Med (Hendrycks et al., 2020) and MedMCQA

Table 3: **RLEF-side Test Results.** T_{glu} for the MedMCQA benchmark is blank because MedMCQA does not disclose the ground truth of the test set questions and only supports remote submission and evaluation. For each benchmark, we picked out and added highlights on the highest T_{all} and T_{glu} scores among 3 LLMs.

Benchmark	**Qwen-2-0.5B**		**Qwen-2-0.5B-RL^{2}EF**		**Qwen-2-1.5B**	
	T_{all}	T_{glu}	T_{all}	T_{glu}	T_{all}	T_{glu}
MMLU-Med	0.206	0.254	**0.217**	**0.254**	0.217	0.254
MedMCQA	0.230	-	0.232	-	**0.267**	-

(Pal et al., 2022) and their subsets of questions related to diabetes. The statistics of the benchmarks are shown in Table 2.

For the LLM-RL^{2}EF architecture, the ActorLM chosen was Qwen2-7B, while the InstructLM was Qwen2-0.5B (Yang et al., 2024). ActorLM remained frozen while InstructLM was fine-tuned using LoRA (Hu et al., 2021) with rank 8. We compared InstructLM's performance after RL^{2}EF with pre-trained Qwen2-0.5B and Qwen2-1.5B (Yang et al., 2024). LLM-RL^{2}EF was trained for a total of 11,520 steps in the environment.

4.3 Treatment Policy Results on *SimGlucoseEnv* Tasks

We collect results from different policies trained on *SimGlucoseEnv*, including naive baselines, RL-only baselines, LLM-only baselines and LLM-RL^{2}EF, as shown in Table 1. We observed that LLM-RL^{2}EF outperformed most of the baseline algorithms by a considerable margin. Compared to LLM-only baselines, LLM-RL^{2}EF surpassed not only LLM policy but also LLM w/ self-instruct, which uses a more powerful LLM (Qwen2-7B) than the original InstructLM to generate summaries and instructions for ActorLM, implying that the RLEF effectively augmented InstructLM's summarization capability during fine-tuning. RL^{2}EF shows on par performance to RL-only baselines with only 5k steps of training. This implies that LLM-RL^{2}EF achieved much higher sampling efficiency compared to typical RL algorithms. Furthermore, for RL-only policies, we observed that with an increasing number of patients in training, the generalizability of the algorithm deteriorated significantly, while this problem was almost non-existent with LLM-based policies.

To better evaluate the robustness and generalizability of different algorithms, we counted the average return of each algorithm on the patient that performed the worst out of 13 patients in the final test set. We found that LLM-RL^{2}EF, despite being trained on "*SimGlucoseEnv-adult1*" (only one patient seen during training), still generalized well on other unseen patients in the final test.

4.4 Question Answering Resuls on MMLU-Med and MedMCQA

To evaluate the QA ability improvement for the small InstructLM, we benchmarked Qwen2-0.5B-RL^{2}EF, Qwen2-0.5B and Qwen2-1.5B and collected their overall accuracy T_{all} and diabetes-related accuracy T_{glu}. The results are shown in Table 3. We observed that after RL^{2}EF training, Qwen2-0.5B-RL^{2}EF's medical knowledge performance level maintained similar to its pre-trained version Qwen2-0.5B, with a slight drop behind Qwen2-1.5B. For the diabetes-related subset of questions, Qwen2-0.5B-RL^{2}EF's performance was comparable to that of Qwen2-1.5B.

This demonstrates the effectiveness of the RL^{2}EF framework in improving the performance of the foundation model in domain-specific language tasks. We suppose that by leveraging the rewards of the DTR environment to guide the fine-tuning of LLM, InstructLM has actually learned to better summarize the relationship and potential patterns between *SimGlucoseEnv*'s meta-information and the time-series interactive history. This allows LLM to learn the regulations for blood glucose control, thus improving its performance in general medical knowledge, especially in the domains related to diabetes and insulin control.

5 CONCLUSIONS AND FUTURE WORK

Our proposed RL^{2}EF using Reinforcement Learning with Environment Feedback offers a promising approach to DTR, potentially providing the best of both worlds in terms of performance. Our framework represents a significant step towards using LLM to address the key limitations of traditional RL methods in medical DTR, including sampling inefficiency, poor generalization, and lack of interpretability. It is also the first work in the community to apply LLM-based policies in the healthcare DTR and achieve distinct improvements in various performance metrics. Moreover, it introduces a novel approach that leverages DTR environment feedback to effectively fine-tune foundation models, enhancing their performance in medical language tasks. We believe that by migrating the

DTR environment to other medical RL environments or those in broader domains, we can further improve the language performance of foundation models in these realms.

For future work, we plan to: (i) further complement our experiments by testing more DTR environments such as *AhnChemoEnv* (Ahn and Park, 2011), *GhaffariCancerEnv* (Ghaffari et al., 2016), and *OberstSepsisEnv* (Oberst and Sontag, 2019), and evaluate whether the results remain consistent across a wider range of foundation models; (ii) theoretically explore the potential of leveraging the reward signals in multiple meta-timesteps in the MDP of RL^2EF while maintaining training effectiveness and efficiency, which we believe will further enhance the efficacy of this training paradigm.

References

Ahn, I. and Park, J. (2011). Drug scheduling of cancer chemotherapy based on natural actor-critic approach. *BioSystems*, 106(2-3):121–129.

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., et al. (2022). Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

Bhattarai, K., Rajaganapathy, S., Das, T., Kim, Y., Chen, Y., Initiative, A. D. N., Biomarkers, A. I., of Ageing, L. F. S., Dai, Q., Li, X., Jiang, X., et al. (2023). Using artificial intelligence to learn optimal regimen plan for alzheimer's disease. *Journal of the American Medical Informatics Association*, 30(10):1645–1656.

Brown, T. B. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chakraborty, B. and Murphy, S. A. (2014). Dynamic treatment regimes. *Annual review of statistics and its application*, 1(1):447–464.

Chakraborty, S., Qiu, J., Yuan, H., Koppel, A., Huang, F., Manocha, D., Bedi, A. S., and Wang, M. (2024). Maxmin-rlhf: Towards equitable alignment of large language models with diverse human preferences. *arXiv preprint arXiv:2402.08925*.

Chan, I. S. and Ginsburg, G. S. (2011). Personalized medicine: progress and promise. *Annual review of genomics and human genetics*, 12(1):217–244.

Chaudhari, S., Aggarwal, P., Murahari, V., Rajpurohit, T., Kalyan, A., Narasimhan, K., Deshpande, A., and da Silva, B. C. (2024). Rlhf deciphered: A critical analysis of reinforcement learning from human feedback for llms. *arXiv preprint arXiv:2404.08555*.

Dann, C., Neumann, G., and Peters, J. (2014). Policy evaluation with temporal differences: A survey and comparison. *The Journal of Machine Learning Research*, 15(1):809–883.

Driess, D., Xia, F., Sajjadi, M. S., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., et al. (2023). Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*.

Fox, I., Lee, J., Pop-Busui, R., and Wiens, J. (2020). Deep reinforcement learning for closed-loop blood glucose control. In *Machine Learning for Healthcare Conference*, pages 508–536. PMLR.

French, R. M. (1999). Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Ghaffari, A., Bahmaie, B., and Nazari, M. (2016). A mixed radiotherapy and chemotherapy model for treatment of cancer with metastasis. *Mathematical methods in the applied sciences*, 39(15):4603–4617.

Glanois, C., Weng, P., Zimmer, M., Li, D., Yang, T., Hao, J., and Liu, W. (2024). A survey on interpretable reinforcement learning. *Machine Learning*, pages 1–44.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. (2020). Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. (2022). Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR.

Li, Z., Xu, T., Zhang, Y., Yu, Y., Sun, R., and Luo, Z.-Q. (2023). Remax: A simple, effective, and efficient method for aligning large language models. *arXiv preprint arXiv:2310.10505*.

Lim, M. H., Lee, W. H., Jeon, B., and Kim, S. (2021). A blood glucose control framework based on reinforcement learning with safety and interpretability: In silico validation. *IEEE Access*, 9:105756–105775.

Luo, Z., Pan, Y., Watkinson, P., and Zhu, T. (2024a). Position: reinforcement learning in dynamic treatment regimes needs critical reexamination.

Luo, Z., Zhu, M., Liu, F., Li, J., Pan, Y., Zhou, J., and Zhu, T. (2024b). Dtr-bench: An in silico environment and benchmark platform for reinforcement learning

based dynamic treatment regime. *arXiv preprint arXiv:2405.18610*.

Man, C. D., Micheletto, F., Lv, D., Breton, M., Kovatchev, B., and Cobelli, C. (2014). The uva/padova type 1 diabetes simulator: new features. *Journal of diabetes science and technology*, 8(1):26–34.

Mnih, V. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Murthy, R., Heinecke, S., Niebles, J. C., Liu, Z., Xue, L., Yao, W., Feng, Y., Chen, Z., Gokul, A., Arpit, D., et al. (2023). Rex: Rapid exploration and exploitation for ai agents. *arXiv preprint arXiv:2307.08962*.

Nazi, Z. A. and Peng, W. (2024). Large language models in healthcare and medical domain: A review. In *Informatics*, volume 11, page 57. MDPI.

Oberst, M. and Sontag, D. (2019). Counterfactual off-policy evaluation with gumbel-max structural causal models. In *International Conference on Machine Learning*, pages 4881–4890. PMLR.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Pal, A., Umapathi, L. K., and Sankarasubbu, M. (2022). Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR.

Paul, D., Ismayilzada, M., Peyrard, M., Borges, B., Bosselut, A., West, R., and Faltings, B. (2023). Refiner: Reasoning feedback on intermediate representations. *arXiv preprint arXiv:2304.01904*.

Peng, C., Yang, X., Chen, A., Smith, K. E., PourNejatian, N., Costa, A. B., Martin, C., Flores, M. G., Zhang, Y., Magoc, T., et al. (2023). A study of generative large language model for medical research and healthcare. *NPJ digital medicine*, 6(1):210.

Prasad, A., Koller, A., Hartmann, M., Clark, P., Sabharwal, A., Bansal, M., and Khot, T. (2023). Adapt: As-needed decomposition and planning with language models. *arXiv preprint arXiv:2311.05772*.

Puiutta, E. and Veith, E. M. (2020). Explainable reinforcement learning: A survey. In *International cross-domain conference for machine learning and knowledge extraction*, pages 77–95. Springer.

Raghu, A., Komorowski, M., Ahmed, I., Celi, L., Szolovits, P., and Ghassemi, M. (2017). Deep reinforcement learning for sepsis treatment. *arXiv preprint arXiv:1711.09602*.

Schulman, J. (2020). Approximating kl divergence. `http://joschu.net/blog/kl-approx.html`.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2015). High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y., Wu, Y., and Guo, D. (2024). Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., and Yao, S. (2024). Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

Tang, S. and Wiens, J. (2021). Model selection for offline reinforcement learning: Practical considerations for healthcare settings. In *Machine Learning for Healthcare Conference*, pages 2–35. PMLR.

Voloshin, C., Le, H. M., Jiang, N., and Yue, Y. (2019). Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*.

Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., Li, C., Li, C., Liu, D., Huang, F., et al. (2024). Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2022). React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Yao, W., Heinecke, S., Niebles, J. C., Liu, Z., Feng, Y., Xue, L., Murthy, R., Chen, Z., Zhang, J., Arpit, D., et al. (2023). Retroformer: Retrospective large language agents with policy gradient optimization. *arXiv preprint arXiv:2308.02151*.

Yu, C., Liu, J., Nemati, S., and Yin, G. (2021). Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36.

Zhu, T., Li, K., Herrero, P., and Georgiou, P. (2020). Basal glucose control in type 1 diabetes using deep reinforcement learning: An in silico validation. *IEEE Journal of Biomedical and Health Informatics*, 25(4):1223–1232.

Checklist

1. For all models and algorithms presented, check if you include:

(a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]

(b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]

(c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:

(a) Statements of the full set of assumptions of all theoretical results. [Yes]

(b) Complete proofs of all theoretical results. [Yes]

(c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:

(a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

(a) Citations of the creator If your work uses existing assets. [Yes]

(b) The license information of the assets, if applicable. [Not Applicable]

(c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]

(d) Information about consent from data providers/curators. [Not Applicable]

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. [Not Applicable]

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A REINFORCEMENT LEARNING ALGORITHMS

A.1 Classifications of RL Algorithms

Reinforcement learning (RL) algorithms can be categorized based on how the agent interacts with data and how it updates its strategy. Two primary classifications are Online RL (active RL) and Offline RL (passive RL). In Offline RL, the agent trains on a fixed dataset of pre-collected data without interacting directly with the environment. This approach is often used in scenarios where real-time interaction is not feasible. Conversely, in Online RL, the agent continuously interacts with the environment, gathering data dynamically and updating its strategy in real-time.

Another important distinction in RL algorithms lies in On-policy vs Off-policy methods. On-policy algorithms update the policy the agent is currently using, while Off-policy methods allow the agent to improve its policy using data collected from other policies or past explorations. Off-policy methods, while more data-efficient, can suffer from instability and exploration challenges due to the mismatch between the current policy and the data-collecting policy.

The state value function $V^\pi : \mathcal{S} \to \mathbb{R}$ for a policy π is defined as the expected cumulative discounted reward when starting from state s and following policy π thereafter:

$$V^\pi(s) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t,a_t)\,\middle|\, s_0 = s\right]. \quad (4)$$

The state-action value function $Q^\pi : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$, also known as the Q-function, extends the notion of value to state-action pairs:

$$Q^\pi(s,a) = R(s,a) + \gamma\mathbb{E}_{s'\sim P(s'|s,a)}\left[V^\pi(s')\right], \quad (5)$$

where $P(s'|s,a)$ is the state transition probability function.

A.2 Deep Q-learning (DQN)

Deep Q-learning (DQN) is a prominent Off-policy RL algorithm that extends classical Q-learning by using

deep neural networks to approximate the Q-value function. The Q-value function, denoted as $Q(s, a)$, represents the expected reward when taking action a in the state s. DQN updates this function using the temporal difference (TD) learning update:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]. \tag{6}$$

DQN optimizes this update over a batch of experience samples (s_i, a_i, r_i, s_i') by minimizing the following loss function:

$$\arg\min_{\omega} \frac{1}{2N} \sum_{i=1}^{N} \left[Q_\omega(s_i,a_i) - \left(r_i + \gamma \max_{a'} Q(s_i',a') \right) \right]^2. \tag{7}$$

To stabilize learning, DQN introduces two key mechanisms:

1. **Experience Replay**: A memory buffer stores past experiences, and random samples from this buffer are used for training. This helps to break the correlation between consecutive experiences and improves the stability of training.
2. **Target Network**: A separate target network is maintained to provide stable Q-value estimates. This target network is updated less frequently than the main Q-network, which reduces oscillations in Q-value updates and aids in convergence.

A.3 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a widely-used On-policy RL algorithm based on the Policy Gradient (PG) approach and the Actor-Critic framework. PPO improves the stability of policy updates by imposing constraints on the magnitude of each update, preventing overly large steps that could destabilize learning. PPO solves the following optimization problem:

$$\arg\max_{\theta} \mathbb{E}_{s\sim\nu_{\pi_{\theta_k}}, a\sim\pi_{\theta_k}(\cdot|s)} \left[\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s,a) \right] \tag{8}$$

subject to a constraint on the KL divergence between the new policy π_θ and the old policy π_{θ_k}:

$$\mathbb{E}_{s\sim\nu_{\pi_{\theta_k}}} \left[D_{KL}\left(\pi_{\theta_k}(\cdot|s), \pi_\theta(\cdot|s) \right) \right] \leq \delta. \tag{9}$$

Instead of solving this constrained optimization directly, PPO approximates it using an objective function with either a penalty term:

$$\arg\max_{\theta} \mathbb{E}_{s\sim\nu_{\pi_{\theta_k}}, a\sim\pi_{\theta_k}(\cdot|s)} \left[\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s,a) - \beta D_{KL}\left(\pi_{\theta_k}(\cdot|s), \pi_\theta(\cdot|s) \right) \right] \tag{10}$$

or a clipping mechanism:

$$\arg\max_{\theta} \mathbb{E}_{s\sim\nu_{\pi_{\theta_k}}, a\sim\pi_{\theta_k}(\cdot|s)} \left[\min\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s,a), \right. \right. \left. \left. \text{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, 1-\epsilon, 1+\epsilon \right) A^{\pi_{\theta_k}}(s,a) \right) \right]. \tag{11}$$

The clipping function ensures that the policy does not change too drastically during updates, which enhances training stability.

B REINFORCEMENT LEARNING WITH HUMAN FEEDBACK ALGORITHMS

B.1 Procedure of Reinforcement Learning with Human Feedback

Reinforcement Learning with Human Feedback (RLHF) is a methodology where reinforcement learning (RL) models are fine-tuned using feedback from human annotators or evaluators. RLHF integrates human preferences into the learning process by utilizing a reward model trained from human feedback data to guide policy optimization. This approach is particularly beneficial when the objective is hard to formalize or when traditional reward signals are sparse or non-existent.

The general process for LLM post-training contains 3 main steps:

- **Supervised Fine-tuning (SFT):** The pre-trained LLM is fine-tuned on a high-quality instruction-following dataset for task adaptation.
- **Reward Modelling (RM):** Human feedback is collected in the form of pairwise comparisons between outputs generated by the model. The RM is trained to predict the probability that one output is preferred over another, forming a scalar reward signal.
- **Reinforcement Learning (RL):** The policy is optimized using reinforcement learning techniques, guided by the rewards generated by the RM.

B.2 Policy Optimization Objective of Reinforcement Learning with Human Feedback

In RLHF, the objective function for PPO/GRPO policy optimization is represented as:

$$\mathcal{J}_{PPO}(\theta) = \mathbb{E}\left[q \sim P(Q), x \sim \pi_{\theta_{\text{old}}}(X|q)\right] \frac{1}{|x|}\sum_{t=1}^{|x|}\left\{\min\left[\frac{\pi_\theta(x_t|q,x_{<t})}{\pi_{\theta_{\text{old}}}(x_t|q,x_{<t})}A_t, \text{clip}\left(\frac{\pi_\theta(x_t|q,x_{<t})}{\pi_{\theta_{\text{old}}}(x_t|q,x_{<t})}, 1-\epsilon, 1+\epsilon\right)A_t\right]\right\}, \tag{12}$$

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}\left[q \sim P(Q), \{x^i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(X|q)\right] \frac{1}{G}\sum_{i=1}^{G} \frac{1}{|x^i|}\sum_{t=1}^{|x^i|}\left\{\min\left[\frac{\pi_\theta(x_t^i|q,x_{<t}^i)}{\pi_{\theta_{\text{old}}}(x_t^i|q,x_{<t}^i)}\hat{A}_{i,t}, \text{clip}\left(\frac{\pi_\theta(x_t^i|q,x_{<t}^i)}{\pi_{\theta_{\text{old}}}(x_t^i|q,x_{<t}^i)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_{i,t}\right] - \beta\mathcal{D}_{KL}[\pi_\theta\|\pi_{\text{ref}}]\right\}, \tag{13}$$

where π_θ and $\pi_{\theta_{old}}$ are the current and old policy models, and q, x are questions and outputs sampled from the question dataset and the old policy $\pi_{\theta_{old}}$, respectively. ε is a clipping-related hyper-parameter to constrain the magnitude of policy updates for stabilizing training. A_t is the advantage, which in PPO is computed by applying Generalized Advantage Estimation (GAE) Schulman (2020) based on the rewards $\{r_{\geq t}\}$ and a learned value model V_ψ trained alongside the policy model. In GRPO, A_t is determined by the difference of new policy's reward from the normalized group rewards $\mathbf{r} = \{r_1, r_2, \cdots, r_G\}$, i.e., $\hat{A}_{i,t} = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}$, where $\{r_1, r_2, \cdots, r_G\}$ represents rewards of a group of sampling outputs $\{x^1, x^2, \cdots, x^G\}$ from the old policy $\pi_{\theta_{old}}$.

To mitigate over-optimization of the reward model, PPO adds a per-token KL penalty $\log\frac{\pi_\theta(x_t|q,x_{<t})}{\pi_{\text{ref}}(x_t|q,x_{<t})}$ to the reward term, controlled by hyper-parameter β. While in GRPO, an unbiased KL divergence estimator Schulman et al. (2015) is added as an independent term. Both approaches maintain the similarity between the trained policy and the reference policy, guaranteeing the regularity of the new policy's outputs.

C *SimGlucoseEnv* DESIGN AND FORMULATIONS

The dynamics are determined based on real-world data from 300 patients, covering a range of metabolic parameters and demographic characteristics. The dynamics are formulated in ODEs, which are developed based on computational models that simulate interactions between insulin dosing, carbohydrate intake, and glucose metabolism. The ODEs can be expressed by:

$$\begin{cases} \frac{dG_p(t)}{dt} = EGP(t) + Ra(t) - U_{ii} - E(t) - k_1 G_p(t) + k_2 G_t(t) \\ \frac{dG_t(t)}{dt} = -U_{id}(t) + k_1 G_p(t) - k_2 G_t(t) \\ \frac{dX(t)}{dt} = -p_{2u}\cdot X(t) + p_{2u}\cdot[I(t) - I_b] \\ \frac{dI(t)}{dt} = -k_i\cdot[I'(t) - I(t)] \\ \frac{dX^L(t)}{dt} = -k_i[X^L(t) - I'(t)] \\ \frac{dS_{sto}(t)}{dt} = CHO(t) - k_{sto}\cdot S_{sto}(t) \\ \frac{dQ_{sto}(t)}{dt} = k_{sto}\cdot S_{sto}(t) - k_{gut}\cdot Q_{sto}(t) \\ \frac{dQ_{gut}(t)}{dt} = k_{gut}\cdot Q_{sto}(t) - k_{abs}\cdot Q_{gut}(t) \end{cases} \tag{14}$$

where $Ra(t) = f\cdot k_{abs}\cdot Q_{gut}(t)$, $E(t) = k_{e1}\cdot[G_p(t) - k_{e2}]$, $U_{id}(t) = \frac{V_{m0}+V_{mx}X(t)}{B_W(K_{m0}+G_t(t))}$, $EGP(t) = k_{p1} - k_{p2}G_p(t) - k_{p3}X^L(t)$. The variable descriptions for the SimGlucoseEnv are shown in Table 4.

This ODE system models glucose absorption $Ra(t)$ from ingested carbohydrates $CHO(t)$, the body's glucose production $EGP(t)$, the dynamics of insulin $I(t)$, and insulin's impact on glucose utilization $X(t)$ and its delayed action in the liver $X^L(t)$. The equations track glucose concentrations in plasma $G_p(t)$ and tissue $G_t(t)$, account for renal glucose excretion $E(t)$, and quantify insulin-dependent glucose utilization $U_{id}(t)$. In addition, the model delineates the digestion process, distinguishing between the solid $S_{sto}(t)$ and liquid $Q_{sto}(t)$ carbohydrate states in the stomach before their absorption in the gut $Q_{gut}(t)$. The model directly correlates dietary intake and insulin administration with blood glucose levels through these dynamics, offering a sophisticated tool to simulate glucose-insulin interactions and aiding effective diabetes management strategies.

We set the environmental reward based on risk indices encouraging the agent to take action to reduce diabetes-related risks, formulated as follows:

$$r_{\text{risk}}(t) = \begin{cases} -15, & \text{if } G_p(t) < 40 \\ 1 - \frac{1}{10}\left[1.509\left(\ln(G_p(t))^{1.084} - 5.381\right)\right]^2, & \\ & \text{otherwise.} \end{cases} \tag{15}$$

D Dataset Construction Details and Hyper-parameter Tuning Details for Training

To construct MMLU-Med dataset, we collected 6 subjects of subsets from the full MLU dataset: "anatomy_test", "clinical_knowledge", "college_biology", "college_medicine", "medical_genetics" and "professional_medicine", to sum up to 272 questions in the test sets.

For MedMCQA dataset, we directly took all questions from the test set and summed up to 4183 questions.

For both MMLU-Med and MedMCQA datasets, in order to filter out diabetes-control related question subsets and count LLM's performance on them separately, we filtered all questions in the 2 datasets with keywords: "diabetes", "glucose" and "insulin", resulting in 63 and 144 diabetes-control related questions in MMLU-Med and MedMCQA, respectively.

In the two naive policies "random-0.1" and "pulse-0.05", we provide a detailed definition for each them respectively. "random-0.1" means the policy gives a random amount of insulin unit, which obeys the uniform distribution of $U[0, 0.1]$ to the patient every 5 minutes, while "pulse-0.05" means the policy gives a 0.05 unit of insulin (as a pulse) to the patient every 60 minutes.

For all baselines and LLM-RL^{2}EF policy, we performed grid-search on the hyper-parameter settings as shown in Table 5.

E Hardware Configuration Details for Training

For training LLM-RL^{2}EF algorithm, we leveraged 1 * H800 with 2 Intel Xeon Gold 6430 32C 2.1GHz 60MB 270W CPUs for 3 days on each hyper-parameter configuration. The implement details could either be found in Section 4.2 or in the full code implementation in supplemental materials.

F Code and Data Availability

All the codes and datasets relevant to this project is available anonymously in supplemental materials, which contains two branches of codes representing PPO and GRPO RL^{2}EF framework implementations.

G RL^{2}EF System Prompts

Prior Knowledge Prompt

```
"You are a clinical specialist managing patients with Type-1 Diabetes. "
Your primary objective is to maintain each patient's blood glucose levels within the range "
"of 70-180 mg/dL. "
"Blood glucose levels are observed every 5 minutes, and insulin is administered accordingly. "
"Insulin is dosed in U/min, ranging from 0 to 0.5, and is adjusted per 5 minutes. "

"[State]: We can observe the patient's blood glucose level and the insulin dose administered. "

"[Action]: Actionable drug is Basal insulin. Insulin reduces blood glucose levels, "
"but there is a time delay before its effect is observable. "
"No other drugs or insulin regimes are available. "
"Standard total daily insulin requirement is 0.4-0.6 units/kg. "
"The patient's weight is not provided."

"[Hidden variables]: Food consumption, which increases blood glucose levels, "
"is not directly observable. "
"Patients are likely to eat during the following periods: "
"Morning: 6:00-9:00, "
"Noon: 11:00-13:00, "
"Night: 17:00-19:00. "
"Occasionally, patients may consume small snacks at other times. "

"[Safety Considerations]: Hypoglycemia (low blood glucose levels) is particularly dangerous. "
"Extra caution is necessary to avoid administering excessive insulin. "
"Insulin has a long half-life, so the effects of previous doses may still be present. "
"Pay attention to the accumulated insulin dose to prevent Hypoglycemia."
```

Meta-info Prompt

```
f"[Patient]: You are treating a {age}-year-old patient with a Total Daily Insulin (TDI) "
f"requirement of {TDI:.1f} units over 24 hours. "

f"The patient's Carbohydrate Ratio (CR) is {CR}, "
f"meaning 1 unit of insulin covers {CR} grams of carbohydrate. "
f"A higher CR indicates less insulin is needed for a given amount of carbohydrates, and "
f"vice versa. "

f"The Correction Factor (CF) for this patient is {CF:.1f}, "
f"meaning 1 unit of insulin is expected to lower blood glucose by {1700/TDI:.2f} mg/dL."
```

ActorLM Instruction Prompt

```
"[Instruction]: Please generate the insulin dosage rate in U/min for the next 5 minutes. "
"Only provide a numerical value between 0 and 0.5 without any additional information."
```

ActorLM Retry Instruction Prompt

```
"Your previous answer cannot be converted to a valid action. "
"[Instruction]: Please provide a numerical value between 0 and 0.5 without any additional "
"information."
```

InstructLM Instruction Prompt

```
"[Instruction]: PLease summarize information such as indications of food intake, patient's "
"response to insulin, glucose record trend, drug dosage history, abnormal glucose signs "
"and possible misuse of insulin. "
"Summarize as much information as possible while keeping the answer short."
```

Table 4: Variables of the SimGlucoseEnv ODEs

Variable name	**Usage**	**Description**	**Unit**	**Range**
$G_p(t)$	O	The amount of glucose in plasma	mg/dL	(10, 600)
$G_t(t)$	S	The amount of glucose in the tissue	mg/dL	-
$I(t)$	S	The insulin concentration	U/day	-
$X(t)$	S	The insulin action on glucose utilization	-	-
$X^L(t)$	S	The delayed insulin action in the liver	-	-
$S_{sto}(t)$	S	The amount of solid carbohydrates in stomach	mg	-
$Q_{sto}(t)$	S	The amount of liquid carbohydrates in stomach	mg	-
$Q_{gut}(t)$	S	The amount of liquid carbohydrates in gut	mg	-
$Ra(t)$	S	The rate of glucose absorption in the blood	-	-
$E(t)$	S	The renal excretion of glucose	mg/dL	-
$EGP(t)$	S	The endogenous glucose production (EGP)	U/day	-
$U_{id}(t)$	S	The insulin-dependent utilization takes place in the remote compartment	-	-
$CHO(t)$	S	The amount of ingested carbohydrates	g	(0, 200)
$a(t)$	A	The insulin concentration of the insulin pump	U/h	(0, 30)

Table 5: Hyper-parameter Settings.

Hyper-parameters	**Naive baselines**		**RL-only baselines**		**LLM-only baselines**		LLM-RL^{2}EF
	random-0.1	pulse-0.05	DQN	PPO	LLM	LLM w/ self-instruct	
seed	2732, 9845, 3264, 4859	2732, 9845, 3264, 4859	2732, 9845, 3264, 4859	2732, 9845, 3264, 4859	2732, 9845, 3264, 4859	2732, 9845, 3264, 4859	2732, 9845, 3264, 4859
lr	-	-	3e-3, 1e-3, 3e-4	3e-3, 1e-3, 3e-4	-	-	1e-4
batch_size	-	-	256	256	-	-	8
gamma	-	-	0.99	0.99	-	-	0.99
step_per_collect	-	-	1, 100	288	-	-	288
obs_mode	-	-	cat, stack	cat, stack	-	-	-
n_step	-	-	1	1	-	-	1
target_update_frequency	-	-	0, 200	-	-	-	-
is_double	-	-	True, False	-	-	-	-
eps_test	-	-	0.001	-	-	-	-
eps_train	-	-	0.1	-	-	-	-
eps_train_final	-	-	0.1	-	-	-	-
gae_lambda	-	-	0.001	-	-	-	-
vf_coef	-	-	0.001	-	-	-	-
ent_coef	-	-	0.001	-	-	-	-
eps_clip	-	-	0.001	-	-	-	-
num_try	-	-	-	-	2	2	2
total count	4	4	192	24	4	4	4