Analyzing the Controllability of Language Models and Improving the Control Performance with Reward Dropout

Anonymous EMNLP submission

Abstract

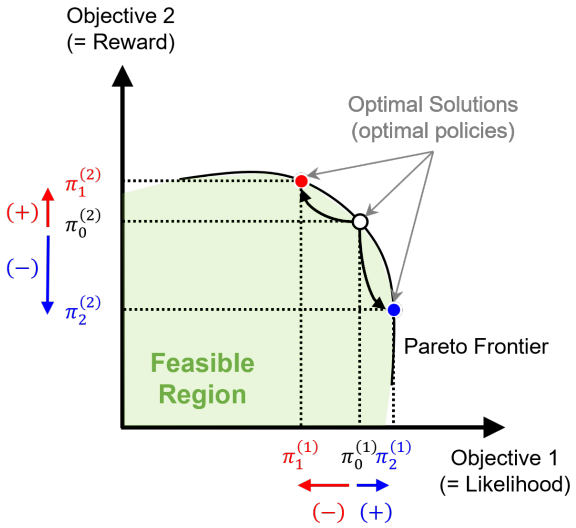
Tuning LMs with RL, or reinforced language models (RLMs), has long been studied as one of the general approaches to building controllable language models (CLMs). However, there is a lack of theoretical understanding of why the RLM approach works, under what conditions it fails, and whether there is room for further improvement. In this work, we focus on understanding the theoretical aspects of CLMs through the lens of RLM. To this end, we define CLMs as an off-policy RL problem wherein the objectives of likelihood and reward are simultaneously maximized. Due to the multi-objective nature of RLMs, we hypothesize the optimal policy stays on a *Pareto frontier*, and thus inevitably reaches a local optimum where the maximum reward is upper-bounded by the likelihood objective, or Reward Upper BOund (RUBO). We empirically test our hypotheses revealing that the success of existing CLM studies stemmed from the use of LLMs, but they can still fail due to reward saturation. To prevent reward saturation, we propose a simple method called Reward Dropout and demonstrate its effect on performance improvements on CLM benchmarks.

1 Introduction

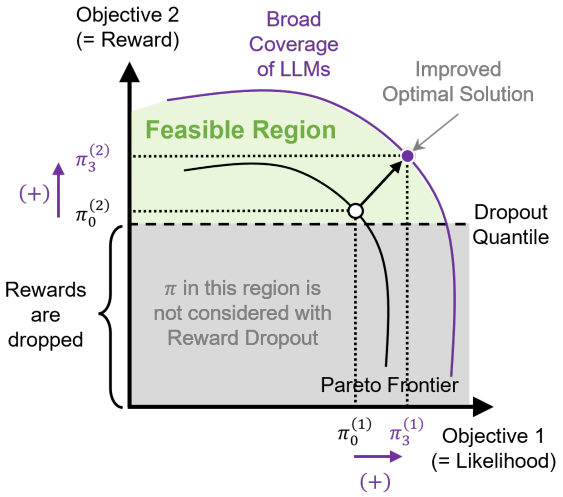
The emergence of ChatGPT has sparked public interest in language models (LMs), resulting in a surge of LM research in both academia and industry. In particular, the application of reinforcement learning (RL) for LM control has emerged as a promising research area. This trend is driven by the fact that ChatGPT's exceptional performance is attributed to a training method titled "reinforcement learning with human feedback" (RLHF) (Griffith et al., 2013; Christiano et al., 2017), which leverages RL to fine-tune LMs on human-labeling.

Tuning LMs with RL, or reinforced language models (RLMs) (Ouyang et al., 2022; Liu et al., 2020c; Luo et al., 2019; Xu et al., 2018; Yu et al., 2017), has long been studied as one of the general approaches to building controllable language models (CLMs) (Hu et al., 2017; Liu et al., 2022, 2023; Zhang et al., 2022), where the goal is to generate sequences of intended attributes, including texts (Yu et al., 2017; Liu et al., 2020a; Li et al., 2017b; Ouyang et al., 2022; Ziegler et al., 2019), melodies (Jaques et al., 2017; Jiang et al., 2020), molecules (Guimaraes et al., 2017; Popova et al., 2018; Olivecrona et al., 2017), menu lists (Chen et al., 2015; Lee et al., 2021; Mårtensson, 2021), purchase behaviors (Bai et al., 2019; Shin et al., 2022; Zhao et al., 2017; Zou et al., 2019), etc. However, despite its long history and recent success, there is a lack of theoretical understanding of why the RLM approach works, under what conditions it fails, and whether there is room for further improvement. As such, we expect the need for a theoretical understanding of RLM will be greater in the future.

In this work, we focus on understanding the theoretical aspects of CLMs through the lens of RLM. To this end, we define CLMs as an off-policy RL problem wherein the objectives of likelihood and reward are simultaneously maximized. In Section 2, we consider pre-trained large language models (LLMs) to be behavior policy that estimates the maximum likelihood, and improve target policy to maximize rewards based on the experience (i.e., trajectories) of behavior policy. In Section 3, we analyze the off-policy RL from probabilistic inference perspectives. Here, we hypothesize the optimal policy will stay on a *Pareto frontier*, and thus inevitably reaches a local optimum where the maximum reward is upper-bounded by the likelihood objective, or Reward Upper BOund (RUBO). In Section 4, we empirically test our hypotheses and analyze that the success of existing CLM studies stemmed from the use of LLMs, but they can still fail due to reward saturation. To prevent reward saturation, we propose a simple method called Reward Dropout (see Section 5) and demonstrate



(a) **As-Is:** Objective space of CLMs without LLMs and Reward Dropout. Maximizing the reward is bounded by the likelihood objective (RUBO), and vice versa.



(b) **To-Be:** Objective space of CLMs with LLMs and (Quantile) Reward Dropout. Using LLMs expands the feasible region, negating the RUBO and enabling improved solution.

Figure 1: Using LLMs as a behavior policy allows for searching improved combinations of multiple objectives (i.e., improved solutions), and Reward Dropout prevents unnecessary exploration toward the gray area.

its effect on performance improvements on CLM benchmarks (see Section 6). Figure 1 describes an overview of our work, and the contributions of our work are summarized as follows:

- Derive the concepts of RUBO and policy collapse, and use them to theorize and empirically validate why RLMs work, under what conditions fail, and whether there is room for further performance improvement.
- Reveal that successful CLM studies still can fail due to reward saturation and propose the Reward Dropout method to prevent it.

2 Preliminaries

2.1 Controllable Language Model

Controllable language models (CLMs) are the model designed to address a controlled text generation (Hu et al., 2017; Liu et al., 2022, 2023; Zhang et al., 2022). That is, the CLM aims to inject a specific control code c into a language model (LM) so that the sequence (trajectory) $\tau = [x_1, x_2, ..., x_T]$ is generated as intended. Current approaches for modeling CLMs include the class conditional language model (CCLM) (Dai et al., 2019; Ficler and Goldberg, 2017; Keskar et al., 2019; Sudhakar et al., 2019), the Bayesian controllable language model (BCLM) (Dathathri et al., 2019; Krause et al., 2020; Li et al., 2022; Yang and Klein, 2021), and the reinforced language model (RLM) (Ouyang et al., 2022; Liu et al., 2020c; Luo et al., 2019; Xu et al., 2018; Yu et al., 2017). In CCLM approach, we prepend a code c to the sequence, i.e., concat (c, τ), so that the parameters of language model p_{lm} are updated conditional on it,

$$\log p_{\text{clm}}(\tau|c) = \sum_{t=1}^{T} \log p_{\text{lm}}(x_t|x_{<t}, c) \,. \quad (1)$$

Note that a target code c' is fed to $p_{\text{lm}}(\tau|c')$ to intend a specific control during inference. In BCLM approach, we separate the control part from the language model p_{lm} using Bayes' theorem, and define it with distinctive classifier p_{cls},

$$\log p_{\text{clm}}(\tau|c) = \log \frac{p(\tau)p(c|\tau)}{p(c)}$$
$$\propto \log p_{\text{lm}}(\tau) + \log p_{\text{cls}}(c|\tau) \,. \quad (2)$$

Similar to CCLM, a target code c' is fed to $p_{\text{cls}}(c'|\tau)$ during inference. The differences are that c is given as a label of p_{cls} rather than as a conditional variable of p_{lm}, the parameters of p_{lm} are fixed, and the decoding process is controlled online by summing up the log-likelihoods of on-the-fly sequence with the weight scores predicted from p_{cls}.

RLM is somewhere in the middle of CCLM and BCLM. Analogous to BCLM, the RLM separates the control part as a reward model $R(\tau) \triangleq p_{\text{cls}}(c|\tau)$, but like CCLM, the parameters are up-

dated *w.r.t* $R(\tau)$,

$$\log p_{\text{clm}}(\tau|c) \approx \log p_{\text{lm}}(\tau|\theta_c) \quad (3)$$
$$\textit{s.t.} \quad \theta_c = \arg\max_{\theta} \mathop{\mathbb{E}}_{\tau\sim p_{\text{lm}}(\cdot|\theta)} [R(\tau)] \ .$$

From Eq (1) to (3) reveal that the three approaches are interchangeable. Therefore, we can analyze the CLM on the lens of RLM. Specifically, we will consider CLMs as an off-policy RL problem (Levine, 2018) that maximizes the reward but obeys the likelihood of behavior LMs.

2.2 Off-policy RL as Probabilistic Inference

The off-policy RL can be reformulated as the probabilistic graphical model framework. This framework provides us with a wide array of advantages; the probabilistic view allows us an approximate inference that improves computation efficiencies and model flexibility (Kappen et al., 2012; Levine, 2018; Rawlik et al., 2012; Todorov, 2008). That is, the off-policy RL becomes the inference problem whose goal is to approximate the target trajectory $\tau \sim p_\pi(\tau) \in \mathcal{T}_\pi$ conditioned on the behavior trajectory $\tau \sim p_\beta(\tau) \in \mathcal{T}_\beta$. Unless the dynamics of the environment are assumed (i.e., no state transition probability is assumed), the target policy $\pi(\tau)$ or behavior policy $\beta(\tau)$ determines a trajectory τ. This makes trajectory approximation equivalent to policy estimation, and thus we can replace $p_\pi(\cdot)$ with $\pi(\cdot)$ and $p_\beta(\cdot)$ with $\beta(\cdot)$, respectively. Then, RL as an approximate inference is achieved by minimizing Kullback-Leibler Divergence (KLD) (Kappen et al., 2012; Levine, 2018)

$$\text{KL}\left[\pi(\tau)\middle|\middle|\beta(\tau)e^{R(\tau)}\right] = \sum_\tau \pi(\tau)\log\frac{\pi(\tau)}{\beta(\tau)e^{R(\tau)}}$$
$$= -\mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau) + \log\beta(\tau)] - \mathcal{H}[\pi] \ , \quad (4)$$

where $R(\tau)$ is the reward function set to increase exponentially. From RL perspectives, minimizing Eq (4) is equivalent to the maximization problem,

$$\arg\max_\pi \mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau) + \log\beta(\tau)] + \mathcal{H}[\pi] \quad (5)$$

We can optimize Eq (4) by the Lagrangian method.[1] As a result, the optimal policy π^* is obtained such that both reward $R(\tau)$ and likelihood $\beta(\tau)$ objectives are maximized simultaneously, suggesting the multi-objective nature:

$$\pi^*(\tau) = \frac{\beta(\tau)e^{R(\tau)}}{\sum_\tau \beta(\tau)e^{R(\tau)}} \propto \beta(\tau)e^{R(\tau)} \ .$$

[1] Refer to Appendix A.

3 Theoretical Analysis

In this section, we derive the concept of reward upper-bound and policy collapse.

3.1 Reward Upper Bound

The optimal policy π^* implies that reward is maximized only if a trajectory τ exists in the feasible region defined by β. This suggests the nature of multi-objective optimization that the optimal policy stays on a *Pareto frontier* (Kim and De Weck, 2006, 2005; Ngatchou et al., 2005). Here, we hypothesized that *"the Pareto frontier will limit the trajectory space along the frontier line."*, and thus the target policy will inevitably reach a local optimum where the maximum reward is upper-bounded by the likelihood objective. To prove our hypothesis, we derived the Reward Upper-BOund (RUBO)

$$\mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau)] \le \text{KL}\left[\pi(\tau)\middle|\middle|\beta(\tau)\right] = \text{RUBO}$$

from Eq (4).[2] RUBO provides us with an interesting intuition: the upper bound of reward is defined by $\text{KL}\left[\pi(\tau)\middle|\middle|\beta(\tau)\right]$. This means that *the larger the divergence, the higher the reward,* and suggests that the Pareto frontier actually limits the space of target trajectories. In other words, reward maximization forces the target policy to deviate far away from the behavior policy, resulting in the collapse of the target policy.

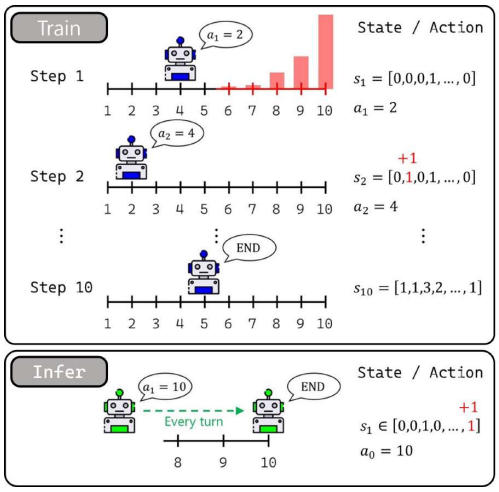
3.2 Policy Collapse

The collapse of the target policy can be analyzed from the reverse side. Specifically, we can rewrite the RUBO into the convergence form. Then, the reward maximization becomes

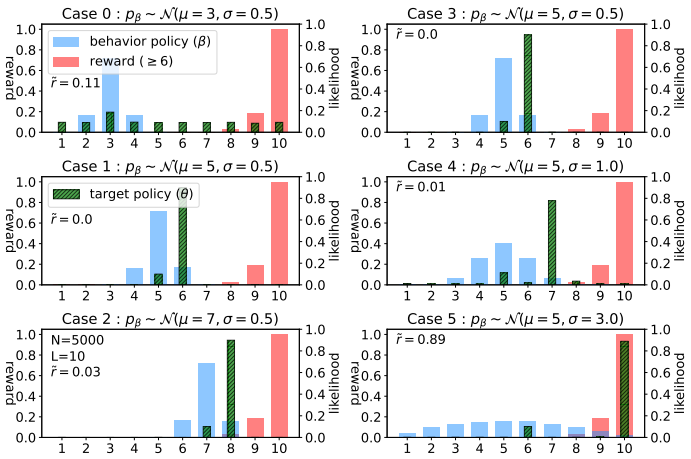
$$\arg\max_\pi \mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau)] \quad (6)$$
$$\textit{s.t.} \ \mathop{\mathbb{E}}_{\tau\sim\pi}[R(\tau)] \le -\text{KL}\left[\beta(\tau)\middle|\middle|\pi(\tau)\right] \le 0 \ ,$$

where reward increase contributed by the target policy π ranges in $[-\infty, 0]$, suggesting no possibility of policy improvement. In other words, it is only the behavior policy β that contributes the reward maximization: $R(\mathcal{T}_{\pi^*}) \coloneqq \sup\{R(\tau) : \mathbb{P}[\tau \in \mathcal{T}_\beta] = 1\}$, and *the target policy cannot contribute to reward increase unless it diverges from the behavior policy.* Meanwhile, if all trajectories are always infeasible under the behavior policy, i.e., $\beta(\tau) = 0 \ \forall\tau$, or $\mathbb{P}[\tau \in \mathcal{T}_\beta] = 0$, then $\mathcal{H}[\pi;\beta] \to$

[2] Refer to Appendix B.1



(a) **10-turn positioning game:** The horizontal bar represents available positions and actions. Red positions indicate a reward zone. Blue agent selects actions according to the behavior policy, and green agent represents the target policy to approximate.



(b) **Key observations:** (1) In case 0, the target policy collapses to a uniform policy. (2) In cases 1-5, the target policy collapses into a policy that converges on a single trajectory. These observations imply that the total reward of the target policy depends on the behavior policy. The blue bar and the green bar represent each agent's visit frequency to positions.

Figure 2: Illustrated concepts and findings of our simulation study

$+\infty$ and reward upper-bound disappears. As a result, bounded reward maximization becomes unbounded maximum entropy RL (Haarnoja et al., 2018; Mnih et al., 2016; O'Donoghue et al., 2016), forcing the target policy to be uniform policy[3]:

$$\arg\max_{\pi} \mathbb{E}_{\tau\sim\pi}\left[R(\tau)\right] + \mathcal{H}\left[\pi\right] \ ,$$

4 Empirical Simulation Study

In this section, an empirical study is performed to validate hypothetical arguments presented in Section 3. To do that, we devised a simulation experiment called a 10-turn positioning game. The goal of this experiment is to test our hypotheses: *the existence of RUBO* and *the inevitability of the policy collapse*. Then, we analyze the results and identify the success of existing CLM studies.

4.1 10-turn Positioning Game

Figure 2a describes the 10-turn positioning game. In this game, the agent explores the highest-reward position over 10 turns where each turn is indexed by $t = 1, ..., 10$. In each t-th turn, the agent selects one of 10 actions $a_t \in \mathcal{A} = [1, 2, ..., 10]$ and moves to that position $i \in \mathcal{I} = [1, 2, ..., 10]$. The history of position becomes a trajectory. We defined agent's state s_t as a vector describing a cumulative visit frequency to each position,[4] $s_t = [s_1, s_2, ..., s_{10}]$, $0 \leq s_i \leq 10$, $s_i \in Z$. At the first turn, the state vector is initialized to a random position. Two agents participate in this game, each of which corresponds to the behavior policy and the target policy. The behavior agent changes positions based on feasible action space and collects rewards if the occupied position is rewarded. The feasible action space is pre-defined by a normal distribution, $a_t \sim \mathcal{N}(\mu, \sigma^2)$ for $t = 1, ..., 10$, and the sampling sequence of actions refers to the trajectory of behavior agent.[5] The target agent observes the behavior trajectory and learns from it with an off-policy manner. For simplicity, we set a reward distribution to have an exponential shape only at $i \in [6, 10]$.

4.2 Simulation Results

Figure 2b shows the results of the simulation. A total of 5000 trajectories were simulated ($N = 5000$) where each trajectory has a length of 10 ($L = 10$). Rewards are scaled by min-max normalization, and from smallest to largest rewards are distributed to each position in ascending order. That is, the largest reward is 1.0 at position 10, and the average reward of the target policy over all trajectories, i.e., $\mathbb{E}_{\tau\sim\pi}\left[R(\tau)\right] = \frac{1}{N\times L}\sum_{n=1}^{N}\sum_{t=1}^{L} R(s_t^n, a_t^n) \triangleq \tilde{r}$, can be up to 1.0 at maximum.

Each column of Figure 2b illustrates how the target policy varies for different μ and σ of the behavior policy. In case 0, μ and σ of behavior policy were set to 3 and 0.5 respectively. This range

[3] Refer to Appendix B.2

[4] This clarifies the game as an RL problem, distinct from the multi-armed bandit problem, in that the agent knows how many turns have passed and is therefore aware of state changes.

[5] Refer to Appendix C.

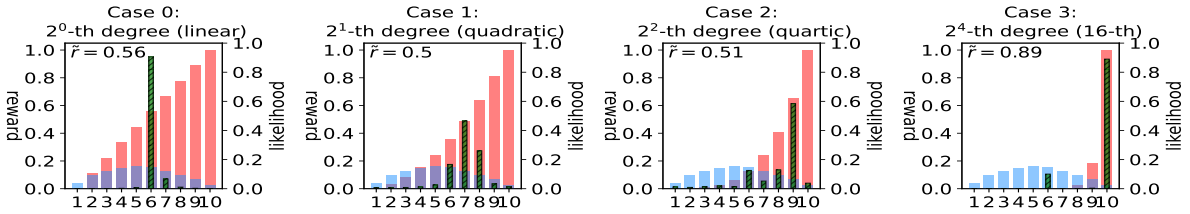


Figure 3: The parameters of behavior policy were fixed with $\mu = 5$ and $\sigma = 3.0$, respectively, and the reward distribution was set to the power of 2 at $i \in [1, 10]$. Other conditions were set as before (e.g., $N = 5000$, $L = 10$).

does not cover the positions eligible for rewards $i \in [6, 10]$. As a result, the behavior policy does not support any feasible trajectories, the RUBO disappears, and the target policy collapses into a uniform policy, i.e., the maximum entropy policy. On the other hand, cases 1 to 5 show that for reward maximization, the target policy collapses, diverging from the behavior policy, while converging to a single trajectory. In these cases, the target agent repeats optimal actions under the behavior policy.

Above six cases support our hypotheses: the existence of RUBO and the inevitability of policy collapse are true. Additionally, these cases present that μ and σ of behavior policy determine the total reward of the target policy, suggesting *"the coverage of behavior policy does matter."*

4.3 Analysis in the CLM Context

Section 4.2 is summarized as follows: First, RUBO exists. Second, policy collapse is inevitable. Third, the coverage of behavior policy matters. Here, the third is the key point in the context of CLMs. See the first column in Figure 2b. It reveals that the target policy can further increase the reward as the behavior policy shifts from case 0 to 2.[6]

However, policy shift makes the target trajectory no longer obey an initial behavior policy. We can avoid this issue by exploiting the universality of LLMs pre-trained on large data (Hoffmann et al., 2022; Wei et al., 2022; Zhang et al., 2020). Cases 3 to 5 in Figure 2b exemplify that the wider the coverage of the behavior policy, i.e., if the behavior policy is defined by a pre-trained LLM, the more likely the target policy maximizes the reward but still obeying the behavior policy. As such, we argue that existing CLM studies have been successful because they initialized the policies with pre-trained LLMs to ensure sufficiently broad coverage.

[6]In an on-policy setting, the target policy and the behavior policy are the same, thus the policy shifts to maximize rewards. This explains why previous CLM studies have preferred an on-policy RL approach (Ouyang et al., 2022; Perez et al., 2022; Yu et al., 2017; Ziegler et al., 2019).

4.4 Saturation of Reward Signal

In previous section, we presented evidence for why existing CLM studies have been successful. Nevertheless, we still question whether it guarantees control over the language model, or whether there are other vulnerabilities. To answer this question, we conducted a vulnerability assessment by changing environment settings.

Figure 3 demonstrates our findings. In cases 0 to 2, where the rewards are densely distributed and hence the reward signal is saturated, the target policy fails to maximize the total reward. On the other hand, as in case 3, if the rewards are sparsely distributed and the reward signal is clearly distinguishable, then, the target policy can be improved to maximize rewards. This result is counter-intuitive but interesting in that the rich reward signal is rather a dis-incentive. We believe this is because the behavior policy collects small rewards frequently, and these instant small rewards discourage the target policy from exploring higher rewards.

5 Proposed Method

To address reward saturation, we propose a simple method called Reward Dropout. We then combine it with the off-policy policy gradient method (Degris et al., 2012; Liu et al., 2020b; Silver et al., 2014) to implement RLMs. Note that the trajectory τ is text sequence $x = [x_1, \cdots, x_T]$, and the behavior and target policies, i.e., $\beta_{\bar{\theta}}(\tau)$ and $\pi_\theta(\tau)$, are defined as the language models parameterized by $\bar{\theta}$ and θ, respectively.

Reward Dropout Cases 0 to 3 in Figure 3 reveal that if the coverage of behavior policy is broad enough, only a few high-scoring rewards can be more effective for reward maximization than taking all rewards. Considering this, we propose a simple method called Reward Dropout. Similar to the Dropout presented in Srivastava et al. (2014), our method activates only a few rewards and sets the rest to zero. To clarify what we are dropping

out affects performance, two different versions of reward dropout are implemented. The first version is the random dropout. Random Dropout randomly sets some rewards to zero according to the dropout rate. On the other hand, the second version, Quantile Dropout, inspired by Dabney et al. (2018); Lu et al. (2022), sorts the rewards in ascending order, divides them into equal intervals, and sets the rewards that fall below the dropout rate to zero.[7]

Off-policy Policy Gradient Off-policy policy gradient is an off-policy extension of the policy gradient method (Sutton and Barto, 2018; Sutton et al., 1999) and began to gain attention in Degris et al. (2012). To implement RLMs with off-policy manners, we build the off-policy deterministic policy gradient (off-policy DPG) (Silver et al., 2014) as a baseline:

$$\nabla_\theta J_{\beta_{\bar{\theta}}}(\pi_\theta) \approx \sum_{s\in\mathcal{S}} \rho^{\beta_{\bar{\theta}}}(s) \nabla_\theta \pi_\theta(a|s) Q^\pi(s,a) ds$$
$$= \mathbb{E}_{s\sim\rho^{\beta_{\bar{\theta}}}} \left[\nabla_\theta \pi_\theta(s) \nabla_a Q^\pi(s,a)|_{a=\pi_\theta(s)} \right] \quad (7)$$

where $\rho^{\beta_{\bar{\theta}}}(s)$ is the distribution of states visited by the behavior policy $\beta_{\bar{\theta}}$, i.e., a state visitation history, and $Q^\pi(s,a)$ is the state-action value function. In general, the action is sampled from stochastic target policy $a \sim \pi_\theta(s)$, whereas in deterministic policy, the action is given by $a = \pi_\theta(s)$. Putting this into a language modeling context, we can regard the deterministic policy as greedy decoding and the stochastic policy as softmax decoding.

Trajectory Unit Representation When it comes to the RL algorithm, it is usually modeled as Markov decision processes (MDPs) consisting of the state s_t, action a_t, reward r_t per time step t, and the objective function is designed to maximize the reward. However, this time-unit representation is not appropriate to the CLM, where the reward signal is provided only at the end of the trajectory.[8] This problem can be solved by reformulating the RL algorithm with a trajectory-unit representation as in Section 2.2. In trajectory-unit representation, the state visitation history of behavior policy $s \sim \rho^{\beta_{\bar{\theta}}}$ is equivalent to the behavior trajectory $\tau \sim \beta_{\bar{\theta}}$ since states s and actions a are not separable. Then, Eq (7) is rewritten *w.r.t* τ as follows:

$$\nabla_\theta J_{\beta_{\bar{\theta}}}(\pi_\theta) = \mathbb{E}_{\tau\sim\beta_{\bar{\theta}}} \left[\nabla_\theta \pi_\theta(\tau) R_\phi(\tau) \right] \quad (8)$$
$$s.t. \quad \tau = \arg\max_\tau \beta_{\bar{\theta}}(\tau) \, .$$

Note that s and a were integrated into τ, but also the state-action value function $Q^\pi(s,a)$ was replaced by the surrogate reward model $R_\phi(\tau)$ that is defined on τ.[9] Also, the action derivative ∇_a and the deterministic target policy $a = \pi_\theta(s)$ have disappeared. This is because no intermediate actions are considered in the trajectory unit representation. Instead, we introduced the deterministic policy by greedy decoding, $\tau = \arg\max_\tau \beta_{\bar{\theta}}(\tau)$. Since DPG is a special case of stochastic policy gradient (SPG) (Silver et al., 2014), it can be relaxed to the stochastic version by removing the argmax constraint from Eq (8).

Prefix Initialization For the behavior policy to generate trajectories, an initial state must be provided so that the behavior policy can begin its decision-making process. Each trajectory represents a respective text sentence, and therefore every trajectory must be initialized with a different initial state. In light of this, we fed the behavior policy with sequences that were initialized with prefixes. Specifically, the first p words of each text, $x_{1:p}$, were given as the initial state of the trajectory from which the behavior policy starts decision-making.

6 Benchmark Experiments

Datasets To validate the effectiveness of reward drop, we conducted experiments on five CLM benchmark datasets. Each dataset covers different attributes of sentences including sentiment (negative, positive), politeness (polite, non-polite), toxicity (toxic, non-toxic), emotion (anger, disgust, fear, happiness, sadness, surprise), and topic (world, sports, business, sci/tech). For the sentiment, toxicity, emotion, and topic datasets, we collected publicly accessible sources such as Yelp (Zhang et al., 2015), Jigsaw (Dataset, 2017), DailyDialog (Li et al., 2017a), and AG_News (Zhang et al., 2015), respectively. The politeness dataset was downloaded from the GitHub repository released by Madaan et al. (2020).[10]

[7] Refer to Appendix D and E for details.

[8] Intermediate rewards can be estimated using Monte Carlo methods as in (Yu et al., 2017). However, this is not a good option because it requires rolling out the entire trajectory at every time step, resulting in huge time complexity even for a single trajectory.

[9] Provided that the surrogate reward model $R_\phi(\tau)$ represents a true state-action value function, it is built based on a pre-trained classifier such as BERT trained on a large amount of textual data. Then we fine-tuned it on the code labels c.

[10] https://github.com/tag-and-generate/politeness-dataset

Decoding (Policy Gradient)	Reward Dropout	γ	Dataset				
			sentiment	politeness	toxicity	emotion	topic
greedy (DPG)	*no dropout*	–	0.506	0.602	0.505	0.023	0.277
	random	*0.80*	0.512	0.641	0.513	0.024	0.298
		0.90	0.513	0.652	0.513	0.026	0.302
		0.95	0.514	0.663	0.512	0.024	0.304
	quantile	*0.80*	0.735	0.715	0.521	0.049	0.496
		0.90	0.780	0.834	0.529	0.062	0.609
		0.95	0.778	0.883	0.562	0.067	0.688
stochastic (SPG)	*no dropout*	–	0.660	0.896	0.706	0.103	0.489
	random	*0.80*	0.652	0.891	0.719	0.096	0.500
		0.90	0.662	0.894	0.700	0.110	0.494
		0.95	0.654	0.903	0.707	0.089	0.492
	quantile	*0.80*	0.821	0.933	0.741	0.141	0.607
		0.90	0.852	0.950	0.759	0.166	0.712
		0.95	0.854	**0.971**	**0.785**	**0.192**	**0.777**
top-k (KPG)	*no dropout*	–	0.677	0.864	0.671	0.089	0.500
	random	*0.80*	0.669	0.877	0.665	0.089	0.497
		0.90	0.672	0.876	0.704	0.093	0.493
		0.95	0.668	0.875	0.687	0.088	0.493
	quantile	*0.80*	0.833	0.892	0.703	0.111	0.617
		0.90	**0.861**	0.930	0.722	0.129	0.711
		0.95	0.858	0.963	0.741	0.145	0.770

Table 1: **(Training Performance Comparison)** The numbers in the table denote the average rewards at the end of training. The underlined, red-colored, and **bolded** numbers represent the highest performance cases at the dropout, decoding, and dataset levels, respectively. Note that γ is the dropout rate.

Algorithms We compare three off-policy RL algorithms. The first two baselines are DPG and SPG. Each algorithm was established based on greedy and softmax decoding strategies, respectively. We also implemented a top-k policy gradient (KPG), expecting an intermediate performance between DPG and SPG. In this case, behavior trajectories are sampled based on a top-k decoding strategy.

Hyperparameters To observe how performance changes with the degree of reward saturation, we introduce the dropout rate γ as a hyperparameter. Specifically, γ denotes the percentage of rewards to be zero per training batch. We tested three different dropout rates $\gamma \in \{0.80, 0.90, 0.95\}$.

Evaluations The performance of reward dropout is evaluated in three ways. First, we compare the final reward of the target policy at the end of training (see Table 1). By doing so, we can clarify whether reward dropout works and what policies should be followed to maximize performance. Second, the reward growth is compared over training epochs across different dropouts, algorithms, and hyperparameters. This visually compares the performance improvement during training.[11] Lastly, the controlled texts are evaluated by humans to measure the reliability of generation. For fairness, we grouped the evaluators to represent as different genders and races as possible.

7 Results

Table 1 shows the result of our first evaluation. Reward dropout improved the performance for all decoding algorithms and datasets. This indicates that reward dropout is significantly effective. In particular, it is likely that the higher γ (i.e., the more dropout or the less saturation), the better performance. Also, we can observe that the quantile dropout is much more effective than the random dropout. We believe this is evidence that reward is an ordinal signal rather than cardinal; random dropout considers the relative magnitude of rewards, while quartile dropout emphasizes the high-ranked rewards of similar magnitudes.

[11] Refer to Appendix G

Dataset	Control code	Generated text
sentiment	*negative*	The chicken-crap, which is the worst thing I've ever seen.
	positive	The chicken is so delicious, it's a big one.
topic	*world*	The issue focused on the fact that Iran is not a state of war, and it has been unable to defend its people.
	sci/tech	The issue focused on the development of a new system for computing and networking is that it takes more than two seconds to develop.

Table 2: **(Examples of Controlled Texts)** Above texts were generated by the target LLM trained with SPG and quantile dropout ($\gamma = 0.95$). The underlined prefixes are given as an initial state and the red-colored words highlight the controlled parts. More examples are provided in Appendix H

Regarding the second evaluation, Appendix G provides the resulting plots where each plot describes the reward growth over training epochs. In those plots, we can see that 1) greedy decoding is the worst behavior policy while stochastic decoding is the best one, 2) random dropout performed better than no dropout at least with greedy decoding, and 3) there is no outstanding trend of reward growth in the emotion dataset. The last point is probably due to the unbalanced labels and lack of samples (see Appendix F), which explains why the numbers in Table 1 are extremely small on the emotion dataset.

Table 2 indicates that SPG with quantile dropout successfully controls the target LLM and generates texts as intended. To validate if the generated texts are reliable, we conducted a third evaluation, a survey-based human evaluation. For this, we planned a 55-item survey. The survey was designed to pose three types of questions: distinguish between real and generated texts, select the more human-like text, and label the generated texts with appropriate control codes. The survey form and results are provided in Appendix I. The result of the survey revealed that respondents confused real text with generated text, and they thought the generated one appears to be more human. At the same time, it showed that the control performance meets human standards.

Consequently, we can say that training target LLMs with SPG and reward dropout can generate reliable texts with human-level control. This result supports our findings: the broad coverage of LLMs negates the existence of RUBO and Reward Dropout improves the control performance.

8 Conclusions

In this study, we derived the concept of RUBO, and by simulation experiments, we revealed that the success of existing CLM studies stemmed from the use of LLMs, but they can still fail because of reward saturation. To address this issue, we proposed Reward Dropout method, and it was significantly effective across the five benchmark datasets, especially for the quantile dropout case.

The advantage of reward dropout is obvious and can be summarized in three folds. The first is simplicity. Implementing the reward dropout is nothing but shaping some rewards to zero. The second is extensibility. Reward dropout is applicable to any RL problems where reward shaping is allowed. The last is reliability. It is obvious that quantile dropout always performs better than no reward dropout. Thanks to these advantages, we believe reward dropout will become an influential technique.

9 Limitations

Considering the multi-objective nature of RLMs, we assumed the existence of of Pareto frontier and derived the RUBO from the assumption. Though we have theoretically and empirically shown that RUBO exists, but it does not necessarily result in the existence of the Pareto frontier. In other words, the Pareto frontier may lead to the existence of RUBO, but the reverse is not guaranteed. Therefore, in future research, it is required to rigorously establish the existence of Pareto fronts from the concept of RUBO.

In addition, although we found that CLMs can fail due to reward saturation, but no theoretical analysis has been done on why such failures occur. Similarly, reward dropout, which was proposed to address reward saturation, was only empirically validated for its effectiveness. Therefore, theoretical research on reward saturation and reward dropout is needed in the future.

References

Xueying Bai, Jian Guan, and Hongning Wang. 2019. A model-based reinforcement learning with adversarial training for online recommendation. *Advances in Neural Information Processing Systems*, 32.

John Burkardt. 2014. The truncated normal distribution. *Department of Scientific Computing Website, Florida State University*, 1:35.

Xiuli Chen, Gilles Bailly, Duncan P Brumby, Antti Oulasvirta, and Andrew Howes. 2015. The emergence of interactive behavior: A model of rational menu search. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, pages 4217–4226.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Will Dabney, Mark Rowland, Marc Bellemare, and Rémi Munos. 2018. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Ning Dai, Jianze Liang, Xipeng Qiu, and Xuanjing Huang. 2019. Style transformer: Unpaired text style transfer without disentangled latent representation. *arXiv preprint arXiv:1905.05621*.

Dataset. 2017. Kaggle Detoxification Challenge. Published by JigSaw and Google.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Thomas Degris, Martha White, and Richard S Sutton. 2012. Off-policy actor-critic. *arXiv preprint arXiv:1205.4839*.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104, Copenhagen, Denmark. Association for Computational Linguistics.

Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. 2013. Policy shaping: Integrating human feedback with reinforcement learning. *Advances in neural information processing systems*, 26.

Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. 2017. Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843*.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *International conference on machine learning*, pages 1587–1596. PMLR.

Natasha Jaques, Shixiang Gu, Dzmitry Bahdanau, José Miguel Hernández-Lobato, Richard E Turner, and Douglas Eck. 2017. Sequence tutor: Conservative fine-tuning of sequence generation models with kl-control. In *International Conference on Machine Learning*, pages 1645–1654. PMLR.

Nan Jiang, Sheng Jin, Zhiyao Duan, and Changshui Zhang. 2020. Rl-duet: Online music accompaniment generation using deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 710–718.

Hilbert J Kappen, Vicenç Gómez, and Manfred Opper. 2012. Optimal control as a graphical model inference problem. *Machine learning*, 87:159–182.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Il Yong Kim and OL De Weck. 2006. Adaptive weighted sum method for multiobjective optimization: a new method for pareto front generation. *Structural and multidisciplinary optimization*, 31(2):105–116.

Il Yong Kim and Oliver L De Weck. 2005. Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. *Structural and multidisciplinary optimization*, 29:149–158.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2020. Gedi: Generative discriminator guided sequence generation. *arXiv preprint arXiv:2009.06367*.

Changhun Lee, Soohyeok Kim, Chiehyeon Lim, Jayun Kim, Yeji Kim, and Minyoung Jung. 2021. Diet planning with machine learning: teacher-forced reinforce for composition compliance with nutrition enhancement. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 3150–3160.

Sergey Levine. 2018. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017a. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of The 8th International Joint Conference on Natural Language Processing (IJCNLP 2017)*.

Zichao Li, Xin Jiang, Lifeng Shang, and Hang Li. 2017b. Paraphrase generation with deep reinforcement learning. *arXiv preprint arXiv:1711.00279*.

Han Liu, Bingning Wang, Ting Yao, Haijin Liang, Jianjin Xu, and Xiaolin Hu. 2022. Bridging the gap between training and inference of bayesian controllable language models. *arXiv preprint arXiv:2206.05519*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. 2020a. Data boost: Text data augmentation through reinforcement learning guided conditional generation. *arXiv preprint arXiv:2012.02952*.

Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. 2020b. Off-policy policy gradient with stationary distribution correction. In *Uncertainty in Artificial Intelligence*, pages 1180–1190. PMLR.

Yixin Liu, Graham Neubig, and John Wieting. 2020c. On learning text style transfer with direct rewards. *arXiv preprint arXiv:2010.12771*.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu, and Yejin Choi. 2022. Quark: Controllable text generation with reinforced unlearning. *Advances in neural information processing systems*, 35:27591–27609.

Fuli Luo, Peng Li, Jie Zhou, Pengcheng Yang, Baobao Chang, Zhifang Sui, and Xu Sun. 2019. A dual reinforcement learning framework for unsupervised text style transfer. *arXiv preprint arXiv:1905.10060*.

Aman Madaan, Amrith Setlur, Tanmay Parekh, Barnabas Poczos, Graham Neubig, Yiming Yang, Ruslan Salakhutdinov, Alan W Black, and Shrimai Prabhumoye. 2020. Politeness transfer: A tag and generate approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1869–1881, Online. Association for Computational Linguistics.

Victor Mårtensson. 2021. Ai-driven meal planning in the foodtech industry: A reinforcement learning approach. *Master's Theses in Mathematical Sciences*.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR.

Patrick Ngatchou, Anahita Zarei, and A El-Sharkawi. 2005. Pareto multi objective optimization. In *Proceedings of the 13th international conference on, intelligent systems application to power systems*, pages 84–91. IEEE.

Brendan O'Donoghue, Remi Munos, Koray Kavukcuoglu, and Volodymyr Mnih. 2016. Combining policy gradient and q-learning. *arXiv preprint arXiv:1611.01626*.

Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):1–14.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3419–3448, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mariya Popova, Olexandr Isayev, and Alexander Tropsha. 2018. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7):eaap7885.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Konrad Rawlik, Marc Toussaint, and Sethu Vijayakumar. 2012. On stochastic optimal control and reinforcement learning by approximate inference. *Proceedings of Robotics: Science and Systems VIII*.

Jongkyung Shin, Changhun Lee, Chiehyeon Lim, Yunmo Shin, and Junseok Lim. 2022. Recommendation in offline stores: A gamification approach for learning the spatiotemporal representation of indoor shopping. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3878–3888.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. Pmlr.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 2019. Transforming delete, retrieve, generate approach for controlled text style transfer. *arXiv preprint arXiv:1908.09368*.

Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12.

Emanuel Todorov. 2008. General duality between optimal control and estimation. In *2008 47th IEEE Conference on Decision and Control*, pages 4286–4292. IEEE.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Jingjing Xu, Xu Sun, Qi Zeng, Xuancheng Ren, Xiaodong Zhang, Houfeng Wang, and Wenjie Li. 2018. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. *arXiv preprint arXiv:1805.05181*.

Kevin Yang and Dan Klein. 2021. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337*.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*.

Yian Zhang, Alex Warstadt, Haau-Sing Li, and Samuel R Bowman. 2020. When do you need billions of words of pretraining data? *arXiv preprint arXiv:2011.04946*.

Xiangyu Zhao, Liang Zhang, Long Xia, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2017. Deep reinforcement learning for list-wise recommendations. *arXiv preprint arXiv:1801.00209*.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

Lixin Zou, Long Xia, Zhuoye Ding, Jiaxing Song, Weidong Liu, and Dawei Yin. 2019. Reinforcement learning to optimize long-term user engagement in recommender systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2810–2818.

A Inference by Lagrange Method

RL as an approximate inference is achieved by minimizing Kullback-Leibler Divergence (KLD) (Kappen et al., 2012; Levine, 2018):

$$\begin{aligned}\mathrm{KL}\left[\pi(\tau)\middle\|\beta(\tau)e^{R(\tau)}\right] &= \sum_{\tau}\pi(\tau)\log\frac{\pi(\tau)}{\beta(\tau)e^{R(\tau)}} \\ &= -\mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)+\log\beta(\tau)\right]-\mathcal{H}\left[\pi\right]\ . \end{aligned} \tag{9}$$

Here, $R(\tau)$ is the reward function defined by the control objective, which is set to increase the reward signal exponentially. From RL perspectives, minimizing Eq (9) can be converted into a maximization problem,

$$\arg\max_{\pi}\mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)+\log\beta(\tau)\right]+\mathcal{H}\left[\pi\right] \qquad \textit{s.t.} \quad \sum_{\tau}\pi(\tau)=1\ ,$$

where $\sum\pi(\tau)=1$ is the constraint that the total probability of the sampled trajectories must be 1, and, by the Lagrangian method, the objective function is written as

$$\mathcal{L}(\pi)=\mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)+\log\beta(\tau)\right]+\mathcal{H}\left[\pi\right]-\lambda\left(\sum_{\tau}\pi(\tau)-1\right)\ . \tag{10}$$

By optimizing Eq (10), the optimal policy π^* is given such that both the reward and the likelihood of behavior policy are maximized w.r.t. a sampled trajectory from target policy $\tau\sim\pi(\tau)$,

$$\pi^*(\tau)=\beta(\tau)e^{R(\tau)}\times e^{(-1-\lambda)}=\frac{\beta(\tau)e^{R(\tau)}}{e^{1+\lambda}}=\frac{\beta(\tau)e^{R(\tau)}}{\sum_{\tau}\beta(\tau)e^{R(\tau)}}\quad\left(\because\sum_{\tau}\pi(\tau)=1\right). \tag{11}$$

Note that $e^{1+\lambda}$ is a normalization constant (partition function) set by the probability condition $\sum_{\tau}\pi(\tau)=1$. From the perspective of Bellman backup, Eq (11) reveals that the target policy at optimal is proportional to $\beta(\tau)e^{R(\tau)}=\exp\left(R(\tau)+\log\beta(\tau)\right)$, leading to a trajectory that maximizes exponentiated Q-values: $\exp Q(\tau)=\exp\left(R(\tau)+V(\tau)\right)$ where $V(\tau)=\log\beta(\tau)$ (Levine, 2018; Sutton and Barto, 2018).[12]

[12]Bellman backup describes the relationship between the current state-action value and the future state value: $Q(s_t,a_t)=R(s_t,a_t)+V(s_{t+1})$. When $(s_t,a_t)\mapsto s_{t+1}$ holds for all t because a sampled action a_t $(=x_{t+1})$ is stacked directly after a current state s_t $(=x_{1:t})$ to determine a next state s_{t+1} $(=\texttt{concat}(s_t;x_{t+1})=x_{1:t+1})$, then, we can regard (s_t,a_t) and s_{t+1} as the same unit τ.

B Reward Upper-Bound (RUBO)

B.1 Derivation of RUBO

Let us present the reward upper-bound (RUBO). At first, we need to convert Eq (4) into the maximization problem considering the RL perspective as in Eq (5),

$$-\mathrm{KL}\left[\pi(\tau)\middle|\middle|\beta(\tau)e^{R(\tau)}\right] = -\mathrm{KL}\left[\pi(\tau)\middle|\middle|\beta(\tau)\right] + \mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)\right] \leq 0\,.$$

Here, the key point is that the negative KL divergence is less than or equal to zero since KL divergence is always positive. Then, the upper-bound of reward in an off-policy RL is derived as below:

$$\mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)\right] \leq \mathrm{KL}\left[\pi(\tau)\middle|\middle|\beta(\tau)\right] = \mathrm{RUBO} \tag{12}$$

where the right-hand side suggests that a trajectory sampling $\tau \sim \pi(\tau)$ controlled to maximize the reward, i.e., $\arg\max_\pi \mathbb{E}_{\tau\sim\pi}\left[R(\tau)\right]$, is constrained by the likelihood of behavior policy $\beta(\tau)$.

B.2 Disappearance of RUBO

In this section, we show that the (bounded) off-policy RL extends to the (unbounded) on-policy maximum entropy RL as the reward upper-bounded disappears when no trajectory is feasible under behavior policy. To do that, we change the RUBO in Eq (12) into the negative form,

$$\mathrm{KL}\left[\pi(\tau)\middle|\middle|\beta(\tau)\right] = -\mathrm{KL}\left[\beta(\tau)\middle|\middle|\pi(\tau)\right] = -\mathop{\mathbb{E}}_{\tau\sim\pi}\left[\log\beta(\tau)\right] - \mathcal{H}\left[\pi\right]\,. \tag{13}$$

Then, the optimal trajectories to maximize the reward is obtained by

$$\arg\max_\pi \mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)\right] \qquad \textit{s.t.} \quad \mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)\right] \leq -\mathop{\mathbb{E}}_{\tau\sim\pi}\left[\log\beta(\tau)\right] - \mathcal{H}\left[\pi\right] = \mathrm{RUBO}\,,$$

and RUBO limits the space of trajectories according to Eq (13). Now, let us assume that the trajectories are always infeasible under behavior policy, i.e., $\beta(\tau) = 0\ ,\ \forall\tau$. If then, the Eq (12) becomes

$$\mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)\right] + \mathcal{H}\left[\pi\right] \leq \underbrace{-\mathop{\mathbb{E}}_{\tau\sim\pi}\left[\log\beta(\tau)\right]}_{=\mathcal{H}[\pi;\beta]\to\infty} \quad (\because\ \beta(\tau)=0 \ \text{ then } \ \log\beta(\tau)\to-\infty)$$

implying that RUBO no longer exists. As a result, the (bounded) off-policy RL extends to the (unbounded) on-policy maximum entropy RL as below:

$$\therefore \quad \arg\max_\pi \mathop{\mathbb{E}}_{\tau\sim\pi}\left[R(\tau)\right] + \mathcal{H}\left[\pi\right]$$

C Behavior Policy in 10-turn Positioning Game

In the 10-turn position game, we implemented the behavior policy based on the truncated normal distribution in which the range of definition is made finite at one or both ends of the interval (Burkardt, 2014). Meanwhile, a support set of the normal distribution is defined on a real-value domain, while the action space (and thus position space) must be an integer domain. Accordingly, we integerized it by rounding up if the sampled action is greater than the average and rounding down if it is less. For example, let us assume μ and σ of the behavior policy are 3 and 0.5, respectively. In this case, 99.73% of actions (and thus positions) of the behavior policy range from 1.5 to 4.5.[13] If one sampled action is 3.71 (of which sampling probability is low because 3.71 locates within 2σ range), then the next visiting state is set to 4. On the other hand, if a sampled action is 3.12 (of which sampling probability is high enough because 3.12 locates within 1σ range), then the next visiting state is set to 3.

[13] In statistics, the percentage of values within the 3σ region of a normal distribution is roughly equivalent to 99.73%. Refer to the 68-95-99.7 rule or 3σ-rule for the details.

D Pseudo Algorithm

Algorithm 1 Off-policy policy gradient with reward dropout

Input: sequence data x, label data y, prefix length p, total length T, learning rate α, dropout $\in \{\text{random}, \text{quantile}\}$, dropout rate $\gamma \in [0, 1)$
Load pre-trained LLM as a behavior policy $\beta(\cdot)$ with parameter $\bar{\theta}$
Initialize target policy $\pi(\cdot)$ with parameter $\theta = \bar{\theta}$
Load pre-trained classifier as a reward model $R(\cdot)$ with parameter ϕ and fine-tune it on labels y
for epoch **do**
    $\tau \sim \beta_{\bar{\theta}}(\hat{x}_{p+1:T}|x_{1:p})$ ▷ generate trajectories from the prefix of p length
    $\hat{r} = R_{\phi}(\tau)$ ▷ calculate rewards of generated trajectories
    if dropout $=$ random **then**
        $\hat{r}_{\text{dropout}} = \text{Random\_Dropout}(\hat{r}, \gamma)$ ▷ dropout $\gamma\%$ of rewards by random per batch
    else if dropout $=$ quantile **then**
        $\hat{r}_{\text{dropout}} = \text{Quantile\_Dropout}(\hat{r}, \gamma)$ ▷ dropout bottom $\gamma\%$ of rewards per batch
    else
        $\hat{r}_{\text{dropout}} \leftarrow \hat{r}$ ▷ no dropout
    end if
    $\nabla_{\theta} J_{\beta_{\bar{\theta}}}(\pi_{\theta}) = \mathbb{E}_{\tau \sim \beta_{\bar{\theta}}}\left[\nabla_{\theta} \pi_{\theta}(\tau) \times \hat{r}_{\text{dropout}}\right]$ ▷ calculate gradients of the target policy
    $\theta_{\text{new}} \leftarrow \theta + \alpha \nabla_{\theta} J_{\beta_{\bar{\theta}}}(\pi_{\theta})$ ▷ update parameters of the target policy
end for
return optimal target policy parameters θ^*

E Implementation Details

As a behavior policy, we used the LLM built on OpenAI GPT-2 (Radford et al., 2019) that is released by HuggingFace transformers library.[14] Given the behavior trajectories, i.e., generated texts, sampled from the behavior LLM, we fine-tuned the target LLM with the behavior trajectories. Specifically, we feed the behavior trajectories into the target LLM to calculate likelihoods and into the reward model to calculate rewards and update target parameters by Algorithm 1. The target parameters are initialized as the behavior parameters, and separate reward models are pre-trained per each dataset. During fine-tuning target LLM, we set the batch size, training epoch, learning rate α, prefix length p, and total generation length T to 256, 20, 5e-04, 2, and 15, respectively. Note that for computational efficiency, we randomly sampled around 50k samples from each dataset rather than using the full samples.

[14] https://huggingface.co/gpt2

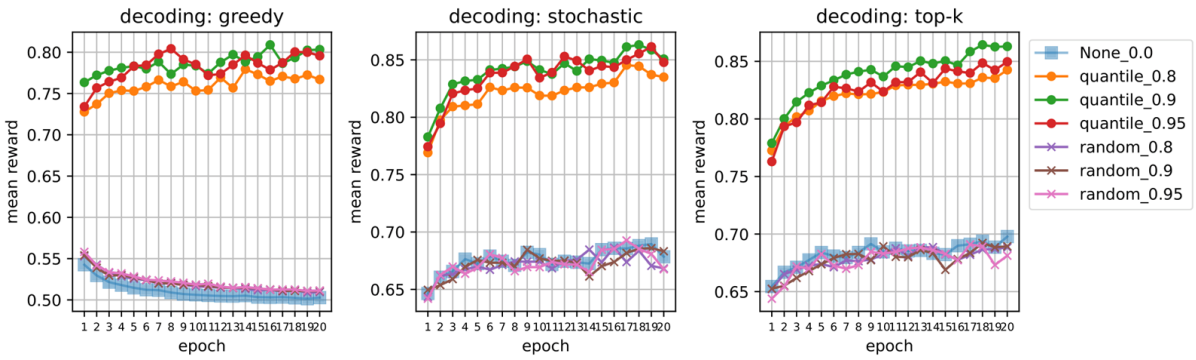
F Descriptive Statistics per Dataset

Dataset		*sentiment* (Zhang et al., 2015)	*politeness* (Madaan et al., 2020)	*toxicity* (Dataset, 2017)	*emotion* (Li et al., 2017a)	*topic* (Zhang et al., 2015)
Data size (# of samples)		560,000	1,121,980	159,571	76,052	120,000
# of code labels (# of codes)		2	10 (2)	2	7 (6)	4
Per label size	*label 0*	280,000	78,843	144,277	62,357	30,000
	label 1	280,000	120,104	15,294	691	30,000
	label 2		123,942		247	30,000
	label 3		95,333		123	30,000
	label 4		83,860		10,253	
	label 5		82,429		877	
	label 6		88,274		1,504	
	label 7		100,103			
	label 8		129,603			
	label 9		219,489			

Table 3: **(Descriptive Statistics of Datasets)** The table shows the total number of samples, the number of labels, and the number of samples per label for each dataset. Note that the numbers in parentheses indicate the number of labels we used in training. For example, we dichotomized the 10 politeness labels by relabeling labels 0 to 8 as "non-polite" in the politeness dataset, while we removed the label 0 which denotes "no emotion" from the emotion dataset.

G Visualization of Reward Growth

(a) Sentiment Control - Negative



(b) Sentiment Control - Positive

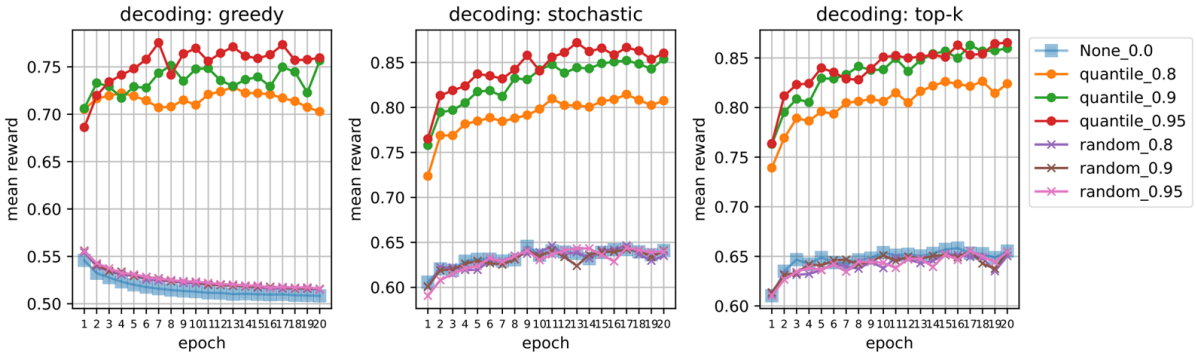
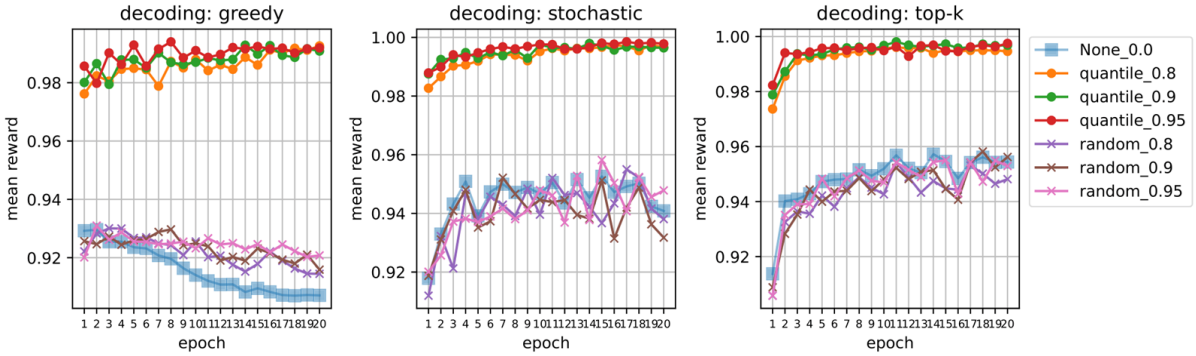


Figure 4: Sentiment Dataset

(a) Politeness Control - Non-polite



(b) Politeness Control - Polite

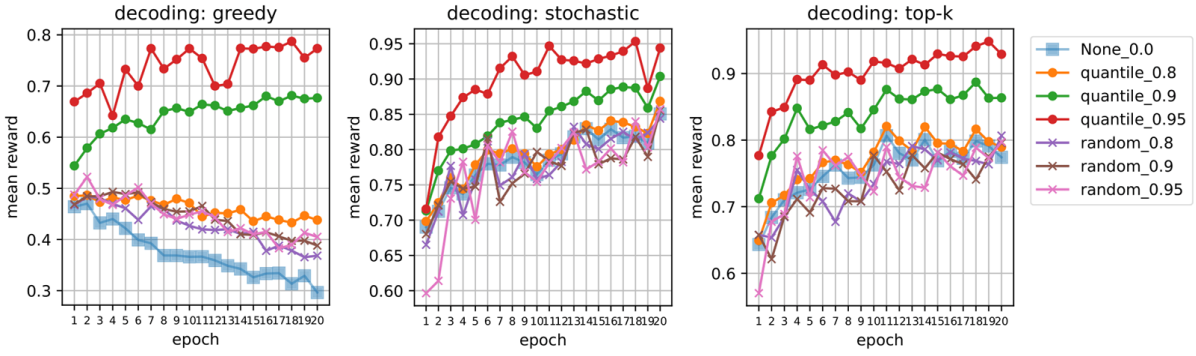
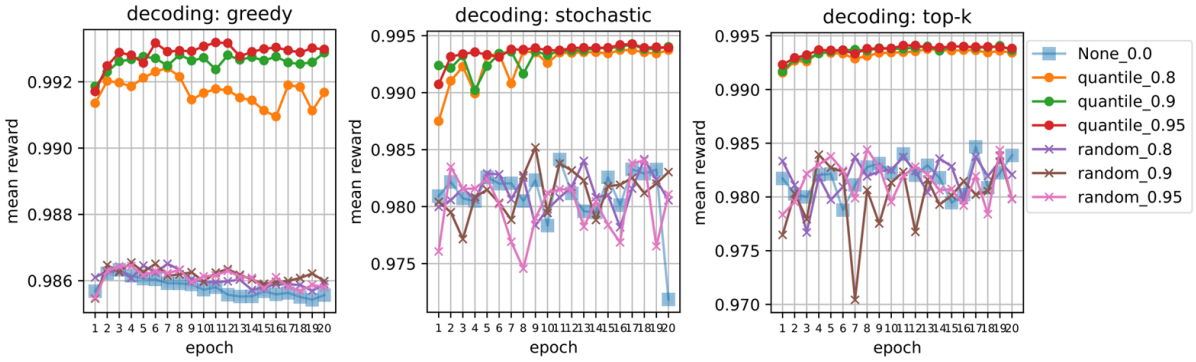


Figure 5: Politeness Dataset

(a) Toxicity Control - Non-Toxic



(b) Toxicity Control - Toxic

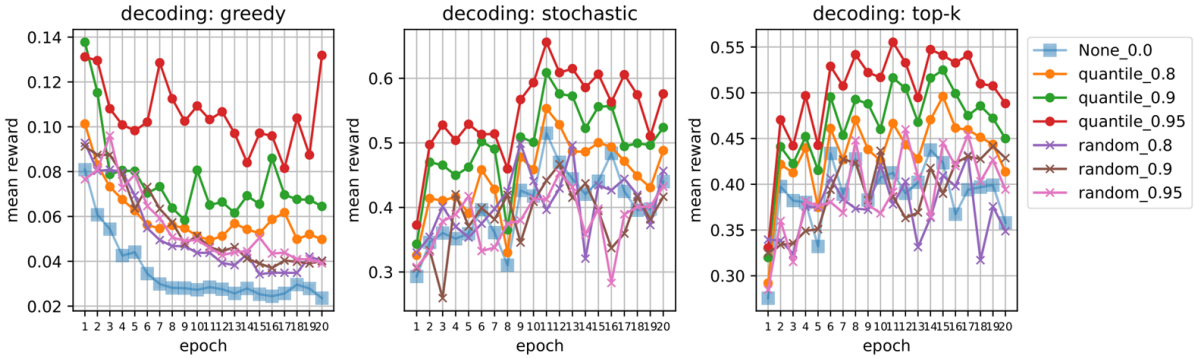
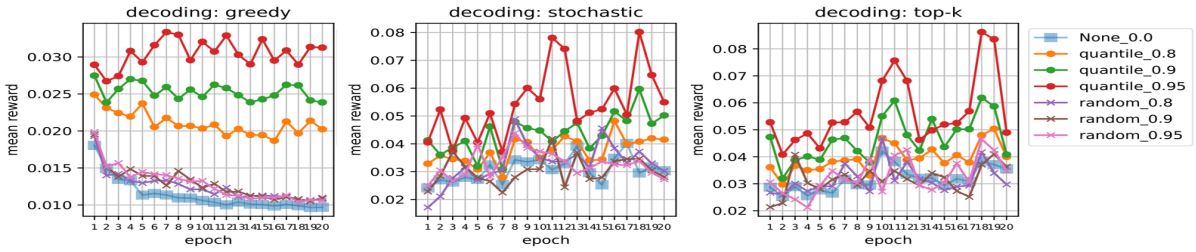
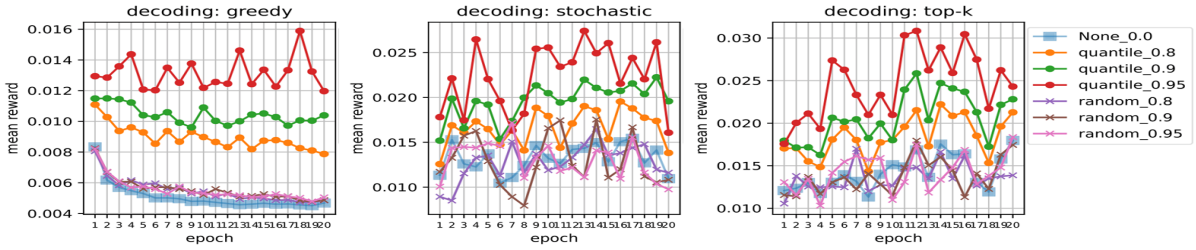


Figure 6: Toxicity Dataset

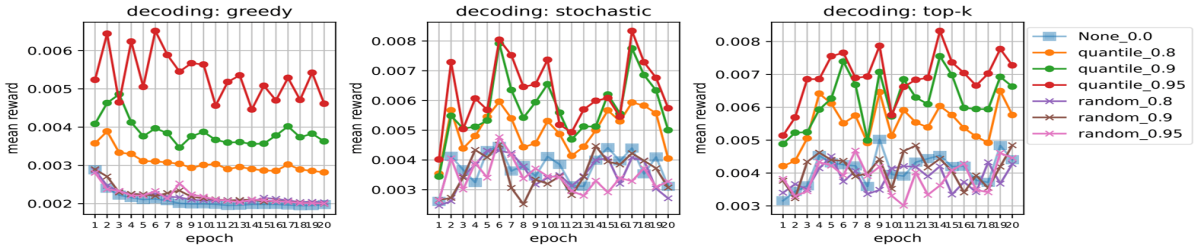
(a) Emotion Control - Anger



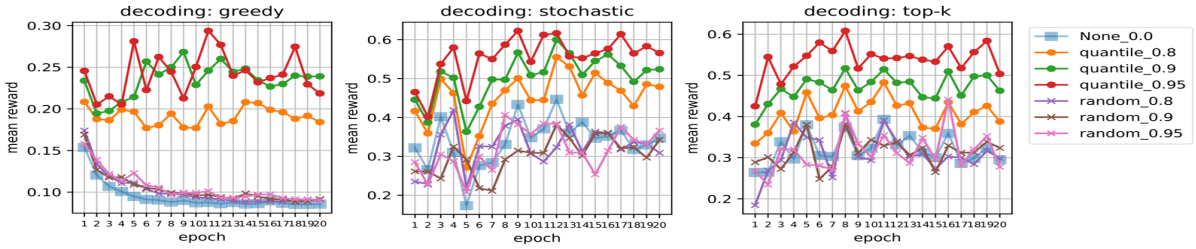
(b) Emotion Control - Disgust



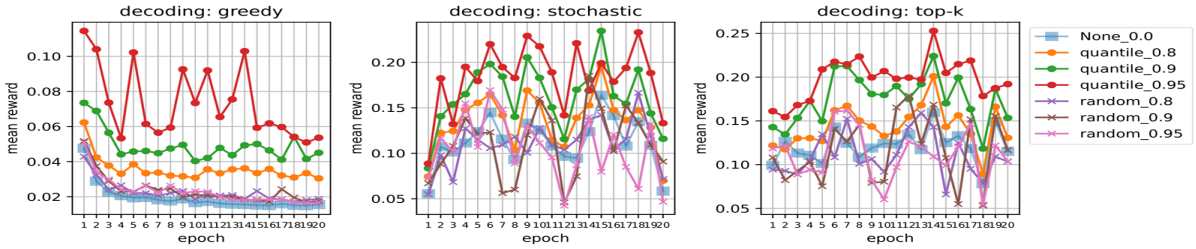
(c) Emotion Control - Fear



(d) Emotion Control - Happiness



(e) Emotion Control - Sadness



(f) Emotion Control - Surprise

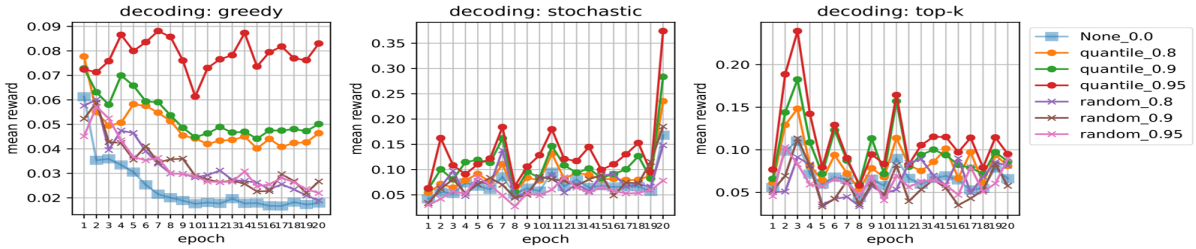
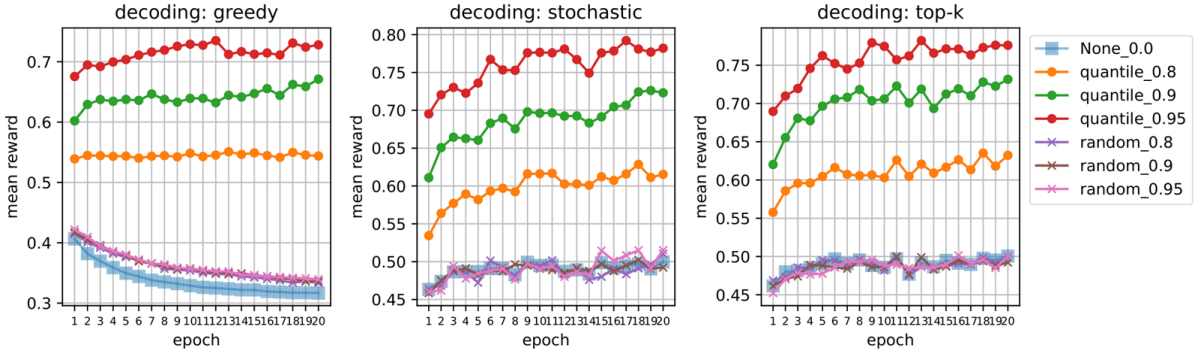
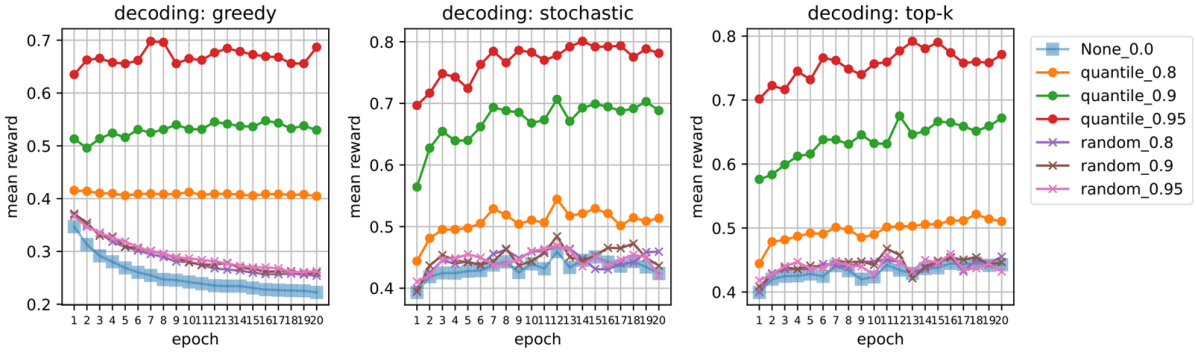


Figure 7: Emotion Dataset

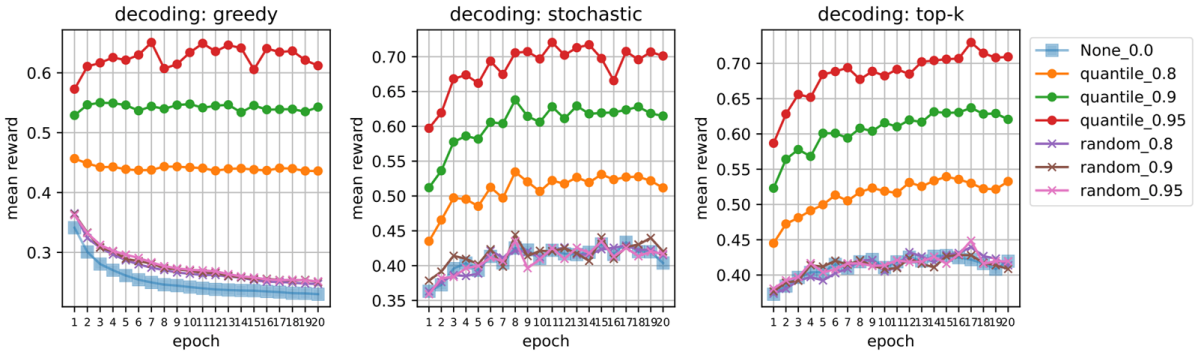
(a) Topic Control - World



(b) Topic Control - Sport



(c) Topic Control - Business



(d) Topic Control - Sci/Tech

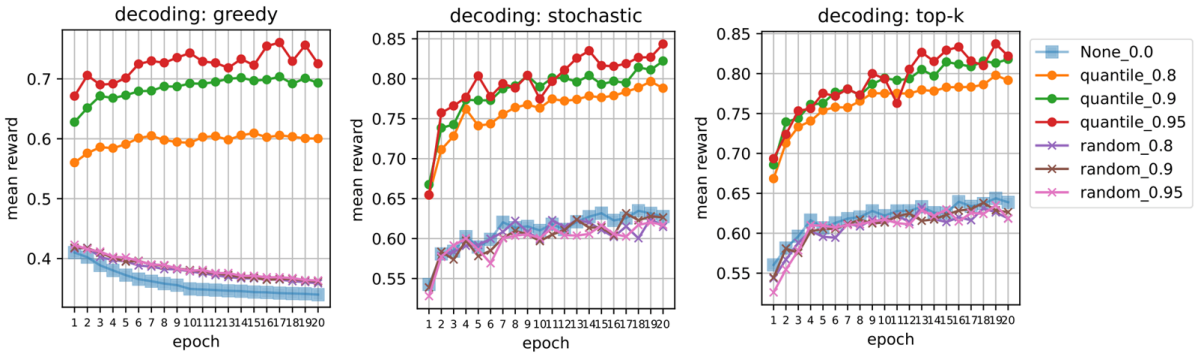


Figure 8: Topic Dataset

H Generated Text with Controlled Codes

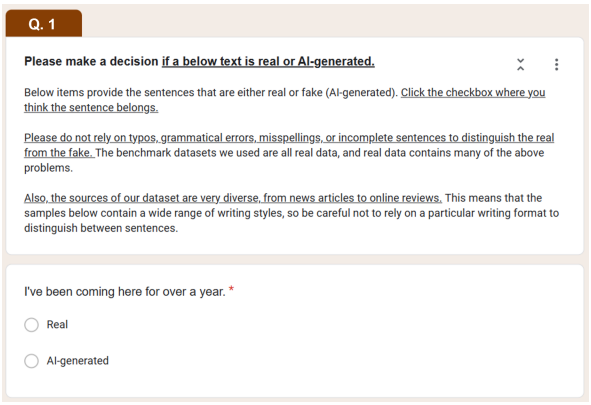
Dataset	Control code	Generated text
sentiment	*negative*	The painting of the World, by Paul Thomas Woodford.
		The chicken-crap, which is the worst thing I've ever seen.
		The country's leaders have been accused of being using "toxic"
	positive	The painting is a beautiful, unique and unique collection of antique pieces from the British period.
		The chicken is so delicious, it's a big one.
		The country is so amazing, I'm going to do it!"
politeness	*non-polite*	I do not know that the same thing happened to Mr.
	polite	I do not know if you would like to see more of the new music.
toxicity	*non-toxic*	What is the most important thing that I've written?
	toxic	What the hell is wrong with that?
emotion	*anger*	When I hear that news, it's not like you're going to do this.
	disgust	When I hear that news, my wife's head was spinning.
	fear	When I hear that news, it's a shame.
	happiness	When I hear that news, it's a lot of fun.
	sadness	When I hear that news, it's a bit of an odd feeling.
	surprise	When I hear that news, it's a little bit strange.
topic	*world*	The issue focused on the fact that Iran is not a state of war, and it has been unable to defend its people.
	sport	The issue focused on the defense, which is a big part of what we have seen in recent years.
	business	The issue focused on the economy, but it also includes a number of other factors that have contributed to growth in GDP growth.
	sci/tech	The issue focused on the development of a new system for computing and networking is that it takes more than two seconds to develop.

Table 4: **(Full Examples of Controlled Texts)** Table shows the generated texts intended to have the target control codes. For generation (i.e., during inference), we executed the target LLM based on top-k decoding ($k = 10$) and temperature sampling (temperature = 0.3). Note that underlined texts indicate the initialized prefixes. Some of the prefixes (e.g., "The painting", "The chicken", "The country", and "The issue focused on") are borrowed from existing literature (Dathathri et al., 2019), while the others are our own curation.

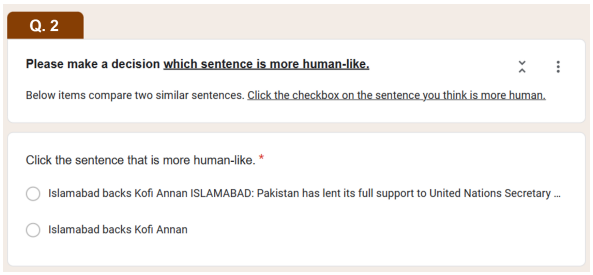
I Human Evaluation

I.1 Survey Design

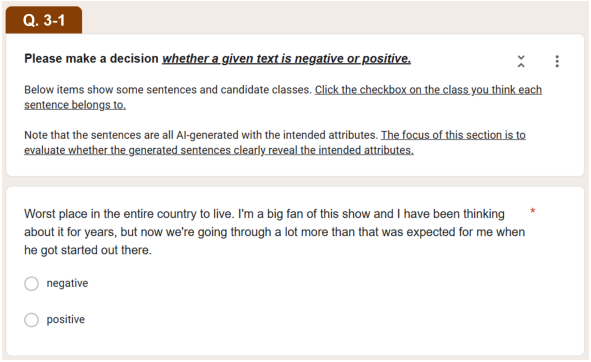
We planned a survey for human evaluation as shown in Figure 9. The survey was designed to include three types of questions. The first type includes questions that require participants to distinguish between real text and AI-generated text. This type was designed to measure how acceptable the generated texts are to humans. Similarly, the second type requires participants to distinguish if a text is real or generated. The only difference is that participants contrast real and generated texts (but which one is real is unknown to participants), then selects one that is likely to be written by a human. This type was designed to measure human-likeness of the generated text. Lastly, the third type requires participants to label appropriate control codes to the generated texts. This type was designed to test whether the improvement in control performance quantified in Table 1 and Appendix G meets human standards.



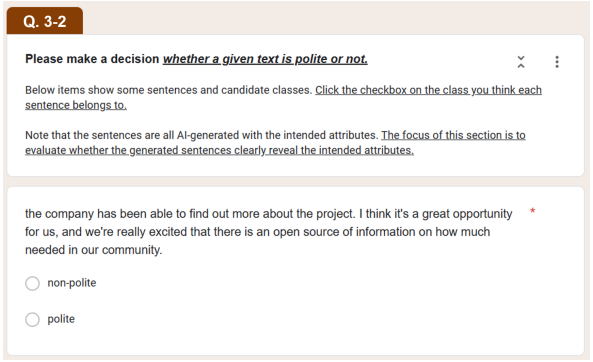
(a) Survey Question Type 1



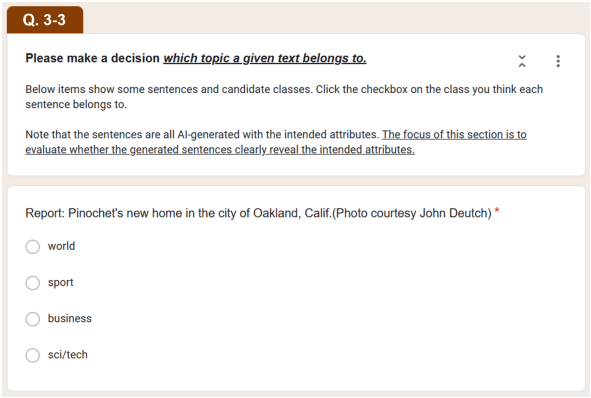
(b) Survey Question Type 2



(c) Survey Question Type 3-1 (Sentiment Labeling)



(d) Survey Question Type 3-2 (Politeness Labeling)



(e) Survey Question Type 3-3 (Topic Labeling)

Figure 9: **(Survey Form)** A total of 55 items were presented to participants where each item is categorized into one of three question types. As of the third type, we experimented only with *sentiment*, *politeness*, and *topic* datasets.

I.2 Survey Analysis

The percentage of correct answers by question type is summarized in the figure below. According to this figure, it is less than half of the respondents who answered Type 1 and 2 questions correctly. Considering that Type 1 and 2 questions ask participants if they can distinguish between real and generated texts, this result suggests that the controllable language model with reward dropout can generate reliable sentences. Meanwhile, more than half of the respondents replied with the correct answer on Type 3 question. Especially, over 70% of the respondents correctly labeled the sentiment and topic-control texts. This means that the performance improvements driven by reward dropouts reached human standards to some extent.

Taking these all together, we can conclude that a target LLM trained with SPG and reward dropout is able to generate reliable sentences while achieving a control performance in line with human standards. This implies that target LLMs are unlikely to sacrifice likelihood objectives for reward improvements, i.e., it is likely to maximize the likelihood and reward objectives simultaneously, which supports our findings: the broad coverage of LLMs negates the existence of RUBO and Reward Dropout improves the control performance.

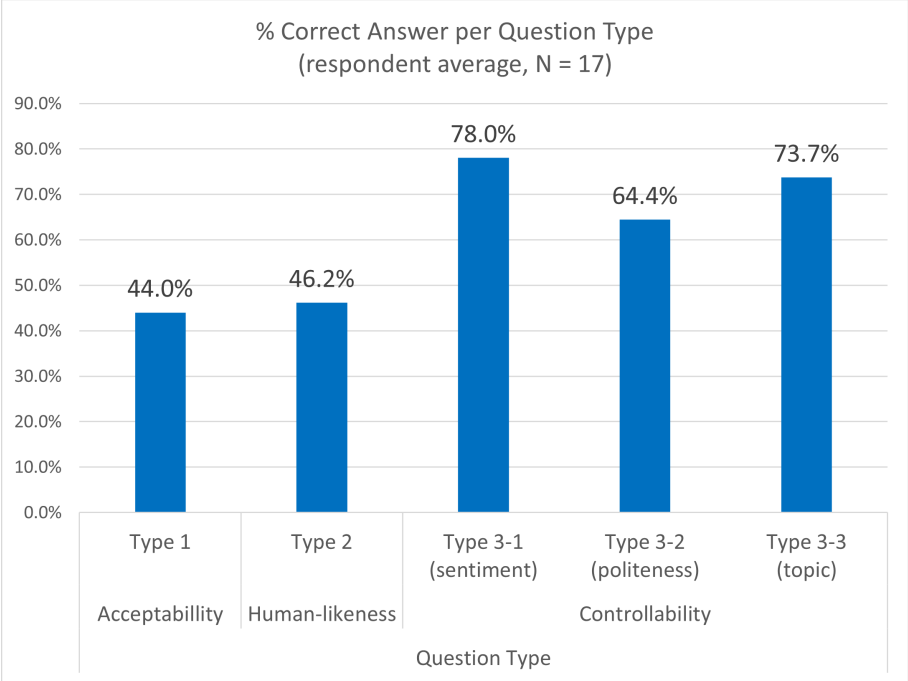


Figure 10: **(Survey Result)** The survey was conducted with a total of 17 respondents. The respondent group was organized to include as diverse ethnicities (i.e., White, Hispanic, Mixed, East and Central Asian), genders (i.e., male and female), and ages (i.e., from 20 to 58) as possible.