An Off-Policy Learning Approach for Steering Sentence Generation towards Personalization

Haruka Kiyohara
Cornell University
Ithaca, NY, USA
hk844@cornell.edu

Daniel Yiming Cao
Cornell University
Ithaca, NY, USA
dyc33@cornell.edu

Yuta Saito
Cornell University
Ithaca, NY, USA
ys552@cornell.edu

Thorsten Joachims
Cornell University
Ithaca, NY, USA
tj@cs.cornell.edu

Abstract

We study the problem of personalizing the output of a large language model (LLM) by training on logged bandit feedback (e.g., personalizing movie descriptions based on likes). While one may naively treat this as a standard off-policy contextual bandit problem, the large action space and the large parameter space make naive applications of off-policy learning (OPL) infeasible. We overcome this challenge by learning a prompt policy for a frozen LLM that has only a modest number of parameters. The proposed **Direct Sentence Off-policy gradient (DSO)** effectively propagates the gradient to the prompt policy space by leveraging the smoothness and overlap in the sentence space. Consequently, DSO substantially reduces variance while also suppressing bias. Empirical results on our newly established suite of benchmarks, called **OfflinePrompts**, demonstrate the effectiveness of the proposed approach in generating personalized descriptions for movie recommendations, particularly when the number of candidate prompts and reward noise are large.[1]

CCS Concepts

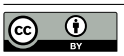
• **Information systems** → **Recommender systems**; *Personalization*; *Language models.*

Keywords

Off-Policy Evaluation and Learning, Sentence Personalization.

ACM Reference Format:
Haruka Kiyohara, Daniel Yiming Cao, Yuta Saito, and Thorsten Joachims. 2025. An Off-Policy Learning Approach for Steering Sentence Generation towards Personalization. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems (RecSys '25), September 22–26, 2025, Prague, Czech Republic.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3705328.3748088

[1]The full version of the paper, including derivation, proofs, and experiment details, is available at **https://arxiv.org/abs/2504.02646**.



RecSys '25, Prague, Czech Republic

ACM ISBN 979-8-4007-1364-4/25/09
https://doi.org/10.1145/3705328.3748088

1 Introduction

As more systems with large language model (LLM)-generated text are starting to become operational, we are naturally collecting increasing amounts of logged user feedback from their system interactions. These feedback signals provide valuable information on whether the generated sentence was effective for the user. Unlike conventional datasets used for LLM training [30], this feedback is available for all users at little cost, providing opportunities for personalizing sentence generation in applications ranging from search and recommendation to improving student motivation in educational chatbots. This creates the need for methods that can use such naturally logged user feedback to enhance the quality and outcome (i.e., reward) of language generation.

To optimize sentence generation, we focus on learning a *prompt policy* (i.e., which prompt to use for a particular user or situation). As detailed in the following, learning a prompt policy is attractive for reasons of (1) safety, (2) cost, and (3) accessibility. With regard to safety, a key requirement for most applications is to not produce harmful outputs [1]. By only adapting the prompts without fine-tuning the LLM itself, we do not run the risk of removing the safety properties of the underlying LLM. Second, compared to the LLM itself, a prompt policy can be a small model, which reduces the required computational resources and the amount of data needed for training [4]. Additionally, compared to using a hand-engineered prompt, a prompt policy enables automated prompt optimization and promises greater personalization. Third, a prompt policy can be trained even in situations where the LLM is closed-weights and available only through an inference API, which makes prompt-policy learning feasible even for small companies or individuals.

While learning a prompt policy is attractive as argued above, using logged user feedback and performing *off-policy learning* (OPL) of a new prompt policy entails several challenges due to the *partial* nature of the feedback. Specifically, the logged data is bandit feedback, containing the reward for only the action (prompt) chosen by the *logging* policy (i.e., the one used in past operations) and not for the other actions that a new policy may choose [13]. This data-generating process is outlined in Figure 1: for each coming user, the logging policy chooses which prompt to use for generating the sentence; then, each user observes only the sentence generated by the chosen prompt and thus reveals the reward (e.g., click) for only this sentence. A naive way to deal with such counterfactuals is to regress the reward and use imputed rewards instead [10, 29, 30].

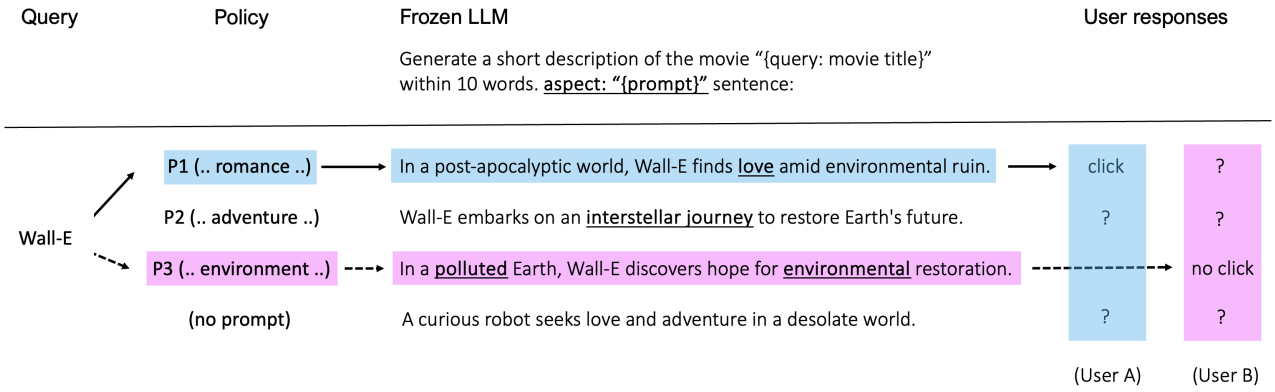


Figure 1: Overview of the prompt-guided sentence personalization with logged bandit feedback. For each incoming user, a policy chooses which prompt to use to generate sentences with a frozen LLM. Each user observes only the sentence generated by the chosen prompt and provides the reward for the corresponding sentence. Logged feedback is partial in that we cannot observe rewards for the sentences generated by prompts *not* chosen by the logging policy. In this example, each prompt (P1, P2, and P3) focuses on the "romance", "adventure", and "environment" aspect, respectively. Examples are generated by ChatGPT-3.5 [2].

However, imputation is often not accurate enough under covariate shift [31] and complex relations between prompts and rewards. An alternative is importance sampling, which re-weighs reward observations w.r.t. the ratio of prompt distribution between the logging and target policies. Nonetheless, this approach suffers from severe variance when the action space is large [27] and bias when the logging policy does not fully explore the action space [21]. These challenges can be particularly problematic in our language generation setting, where we need to deal with a rich and diverse set of candidate prompts as actions.

The key shortcoming of the standard approaches lies in treating each prompt independently, not taking the information about the generated sentence into account. In response, **this paper explores and presents a method to leverage the similarity among generated sentences to make large-scale OPL for prompt-guided sentence generation efficient and tractable**. Specifically, our *Direct Sentence Off-policy gradient* (DSO) estimates the policy gradient in the sentence space (i.e., not in the action space of prompts) to take the generated sentence into account. We enable this by applying the importance weight in the (marginalized) sentence space and by re-sampling actions conditioned on the sentence when calculating the score function. Since marginalization contributes to reducing the scale of the importance weights and since re-sampling works as an implicit data augmentation about the prompt, we can achieve variance reduction of DSO compared to typical OPL methods. Moreover, by aggregating similar prompts and sentences using kernels, DSO also keeps the bias small.

Finally, we conduct experiments on a newly developed OPL benchmark suite, called **OfflinePrompts**, in both synthetic and full-LLM settings. The results demonstrate the effectiveness of our approach, particularly when the number of prompts is large, showing a **5x increase in the performance in the full-LLM experiment**. We also provide our benchmark suite, including the full-LLM environment for generating personalized movie descriptions for recommendations based on the MovieLens [9] dataset, as an open-source resource. Its easy-to-use API will accelerate both future research and practical applications of OPL of prompt-guided sentence generation with logged feedback.

Our contributions are summarized as follows.

- **Problem formulation**: We present prompt-policy learning from logged feedback as OPL of contextual bandits, providing a new pathway for personalized sentence generation.
- **Tailoring OPL methods for language generation**: We identify how to effectively leverage similarity in sentences by deriving the gradient directly in the sentence space.
- **Benchmarks and open-source**: We conduct extensive experiments and provide a new open-source benchmark suite for future research and applications.

2 Related Work

This section summarizes the most closely related work.

Prompt Tuning. Prompt tuning is a cost-efficient approach for optimizing sentence generation for some specific downstream applications, including translation, summarization, and sentiment analysis. Unlike fine-tuning, which requires the updates of the millions of parameters of the LLM itself, prompt tuning reuses a "frozen" pre-trained LLM and optimizes only the choice of the special tokens added to the original input sentence called *prompts* [2]. Deng et al. [4] presented a reinforcement learning (RL) formulation of prompt tuning, which optimizes the prompts via policy gradient by treating a frozen LLM as a black-box reward generator. While this formulation is relevant to ours, the critical limitation of Deng et al. [4] and similar online exploration papers [6] is to assume that feedback (i.e., reward) is easily accessible. Unfortunately, such an assumption is unrealistic in many real-world applications where online interactions with users can be costly, harmful, or sometimes even unethical [8, 18]. Instead, we present a way to leverage logged user feedback naturally collected through past operations.

Reinforcement Learning for Language Generation. RL from Human Feedback (RLHF) is a widely-studied approach to align the output of LLMs using human annotation [3, 17, 19, 30]. Specifically,

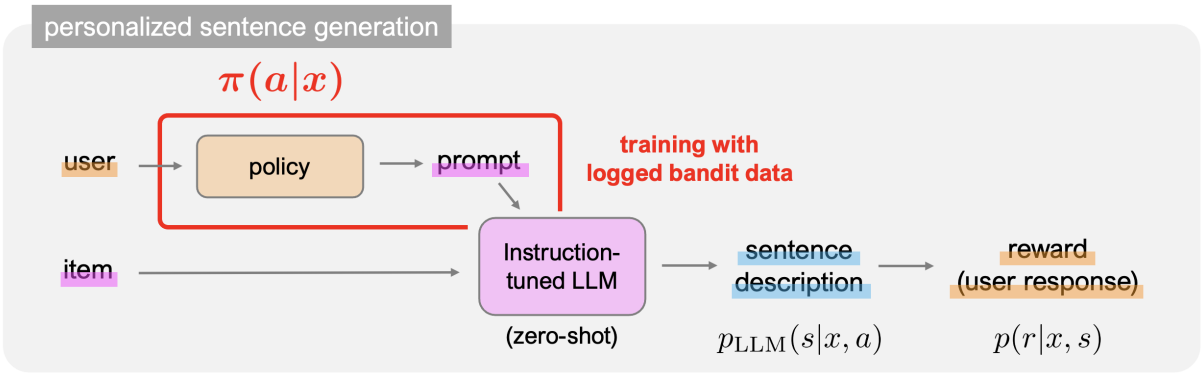


Figure 2: Procedures of the personalized sentence generation. A policy aims to identify a suitable prompt (e.g., genres of the movie or other complex prompts) for each user so the generated sentence aligns with the user-dependent preference.

RLHF asks human annotators to compare two sentences and provide labels to indicate which sentence is more appropriate for a downstream task (e.g., translation). Then, using the pairwise feedback, RLHF trains a model to predict the task-specific score of each sentence to preserve the preference. The key challenges of RLHF are twofold: (1) RLHF incurs substantial cost and ethical concerns for human annotation (e.g., lack of pairwise preference data) [1, 16] and (2) monitoring if annotators provide sufficiently reliable labels for RLHF, as done in [19, 30], can be difficult when preferred sentences change among annotators in tasks related to personalization. OPL from logged bandit feedback naturally resolves the above difficulties of annotation for RLHF.

Off-Policy Evaluation and Learning. Off-Policy Evaluation and Learning (OPE/OPL) studies how to use naturally collected user feedback to evaluate and learn new contextual bandit or RL policies [7, 24]. Regression-based and importance sampling (IS)-based approaches are prevalent in OPL. First, the regression-based approach trains a reward predictor and then optimizes a new policy using imputed rewards. While this approach performs well when the reward predictor is accurate for the entire action (i.e., prompt) space, such an accurate regression is often demanding due to the issues such as counterfactuals and covariate shift [31]. In contrast, the IS-based approach aims to estimate the policy gradient unbiasedly from actual reward observations by correcting the distribution shift [20]. However, IS often suffers from high variance and deficient support, particularly when the action space is large [14, 22, 25, 26]. To overcome the limitations of naive approaches, Saito et al. [27] has recently proposed a two-stage OPL framework called POTEC, which first chooses which cluster among pre-defined action-clusters to use by applying cluster-wise IS and then chooses which action within the chosen cluster to use. However, good clusterings are often hard to identify, and this approach discards information about the generated sentences. In response, we present a way to leverage similarity among sentences by estimating the policy gradient directly in the sentence space. Another related literature is Kallus and Zhou [12], which discuss OPE of deterministic policies in a continuous action space. While we share the ideas of using kernels with Kallus and Zhou [12], our idea comes from a different notion of deriving the gradient directly in the (marginalized) sentence space. Moreover, the theoretical analysis is entirely different, as we apply kernels to the logging policy, while Kallus and Zhou [12] do not. Finally, our new benchmark, which simulates the personalized generation of sentences, is a unique contribution of ours to the OPE/OPL community.

3 Problem Formulation

We start by formulating prompt optimization as a new type of OPL problem, which we call *contextual bandits with auxiliary outputs*.

Let $u \in \mathcal{U} \subseteq \mathbb{R}^{d_u}$ be a d_u-dimensional user feature vector (e.g., demographic profile or user id), sampled from an unknown distribution $p(u)$. Let $q \in \mathcal{Q} \subseteq \mathbb{R}^{d_q}$ be a *query* (e.g., query to a frozen LLM), sampled from a conditional distribution $p(q|u)$. Let $a \in \mathcal{A}$ be a (discrete) *prompt*, where each prompt is associated with some vectorial embedding, $e_a \in \mathbb{R}^{d_e}$, where d_e is the dimension of the embeddings. The prompt is used to generate a sentence via a frozen LLM. This process can be formulated as a procedure of sampling sentence $s \in \mathcal{S}$ as an auxiliary output from the stochastic output distribution of the LLM: $p_{\mathrm{LLM}}(s|q,a)$. A user will respond to the output sentence and provide some reward $r \in \mathbb{R}$ (e.g., click, like, or purchase), where r follows $p(r|u,q,s)$. Let $\pi \in \Pi$ be a *prompt policy* where $\pi(a|u,q)$ is the probability of choosing *prompt* a for *context* $x := (u,q) \in \mathcal{X}$. Our goal is to optimize the prompt policy to maximize the expected reward, defined as

$$V(\pi) := \mathbb{E}_{\underbrace{p(u)p(q|u)}_{=p(u,q)}\pi(a|u,q)\underbrace{p_{\mathrm{LLM}}(s|q,a)p(r|u,q,s)}_{=p(r,s|u,q,a)}}[r]$$
$$= \mathbb{E}_{p(x)\pi(a|x)p(r,s|x,a)}[r].$$

We describe the whole sentence generation process in Figure 2.

When running a prompt policy π_0 ($\neq \pi$) as part of an operational system, it works as a *logging* policy and generates logged feedback of the following form:

$$\mathcal{D} := \{x_i, a_i, s_i, r_i\}_{i=1}^n \sim \prod_{i=1}^n p(x)\pi_0(a|x)p_{\mathrm{LLM}}(s|x,a)p(r|x,s)$$

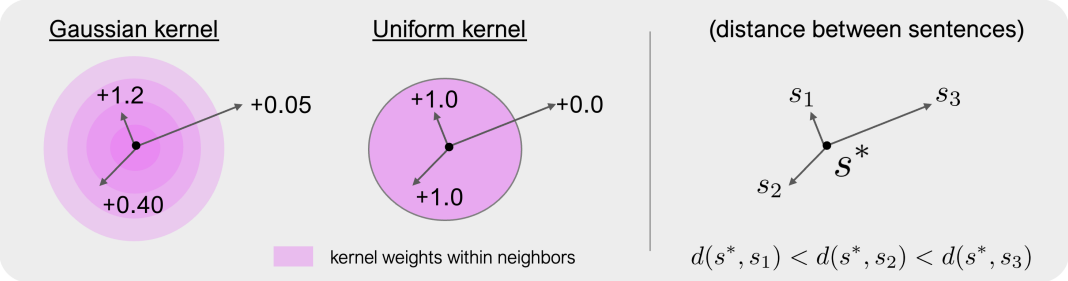


Figure 3: Examples of the kernel weights and (soft) rejection sampling in the marginalized sentence space. DSO implicitly augments the data to take the observations for the neighboring sentences into account and propagates the gradient to prompts that generate neighboring sentences. In this example, (Left) uses a smooth kernel like a Gaussian kernel, and (Right) uses a piecewise constant kernel like a uniform kernel.

where n is the data size and i is its index. The logged data informs us whether the prompt (a_i) results in a high reward or not (r_i) for a particular context (x_i). However, a difficult aspect of using the logged data is that the reward observation is *partial*, i.e., it is observed only for the prompt chosen by the logging policy (π_0) but not for all the other actions. This can be particularly challenging when training a new policy π on the logged data, as π may choose actions that are not chosen by π_0. Thus, we need to address such *counterfactuals* and *distribution shift* between the logging and learning policies when using logged data for a reliable policy optimization [31].

In the rest of the paper, we parameterize the policy as π_θ using some parameters $\theta \in \Theta$ (e.g., a neural network). We also define $R(x,a) := \mathbb{E}[r|x,a]$ and $R(x,s) := \mathbb{E}[r|x,s]$. Finally, $z \sim p(z)$ indicates that we sample a single random variable z from the probability distribution $p(\cdot)$ to simulate monte-carlo samples, for any random variable z and its probability distribution.

3.1 Conventional approaches

We first review direct applications of typical OPL methods and discuss their limitations.

Regression [15]. A typical way of using logged data is to train a reward predictor $\hat{R}$ [29, 30], and then use the predicted reward to estimate the policy gradient (PG)[2].

$$\nabla_\theta V(\pi_\theta) \approx \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{a\sim\pi_\theta(a|x_i)}\left[\nabla_\theta \log \pi_\theta(a|x_i)\hat{R}(x_i,a)\right].$$

Oftentimes, an accurate regression for OPL is difficult to obtain when the relation between prompts and reward is complex. This is because the reward observation is partial and covariate shift arises between the logging policy (π_0) and the target policy (π_θ). If the learned regression model $\hat{R}$ is inaccurate, the estimated PG can be heavily biased [31].

Importance sampling (IS) [31]. Instead of using potentially inaccurate regression, IS corrects the distribution shift between π_0 and π_θ by reweighing the observations:

$$\nabla_\theta V(\pi_\theta) \approx \frac{1}{n}\sum_{i=1}^{n} \frac{\pi_\theta(a_i|x_i)}{\pi_0(a_i|x_i)}\nabla_\theta \log \pi_\theta(a_i|x_i) r_i.$$

IS is unbiased under the *action support* condition, i.e., $\forall (x,a) \in \mathcal{X}\times\mathcal{A}$, $\pi_\theta(a|x) > 0 \implies \pi_0(a|x) > 0$. However, IS produces considerable bias due to the violation of the condition (deficient support) [21] and extremely high variance due to large importance weight [22, 26, 27], which are likely when the action space is large. The key shortcoming here is that the typical methods treat each prompt independently and discard the rich information about the generated sentence when estimating the policy gradient.

4 Direct Sentence Off-Policy Gradient (DSO)

The key idea is to make the most of the information about the generated sentence by **taking the policy gradient directly in the sentence space** as follows.

$$\nabla_\theta V(\pi_\theta) = \mathbb{E}_{p(x)\pi_\theta(s|x)}[\nabla_\theta \log \pi_\theta(s|x) R(x,s)].$$

Even when we parameterize the policy in the prompt space, deriving this gradient is possible because we can write the sentence distribution and the score function as $\pi_\theta(s|x) = \sum_{a\in\mathcal{A}} p_{\mathrm{LLM}}(s|x,a)\pi_\theta(a|x)$ and $\nabla_\theta \log \pi_\theta(s|x) = \mathbb{E}_{\pi_\theta(a|x,s)}[\nabla_\theta \log \pi_\theta(a|x)]$, respectively. However, one potential concern of this approach is that we may suffer from data sparsity when estimating the gradient for each sentence s, as sentences are high-dimensional. Thus, we further consider taking the gradient in the **marginalized sentence space** to enable data-efficient OPE as

$$\nabla_\theta V(\pi_\theta) = \mathbb{E}_{p(x)\pi_\theta(\phi(s)|x)}[\nabla_\theta \log \pi_\theta(\phi(s)|x) R^{\pi_\theta}(x,\phi(s))],$$

where $\phi(s) \in \Phi(\mathcal{S})$ is the kernel-based neighbors of sentence s. Its probability density, policy distribution, and expected reward are defined as follows.

- $\mathbb{P}(\phi(s)|\cdot) := \int_{s'\in\mathcal{S}} K(s',s;x,\tau)\mathbb{P}(s'|\cdot)ds',\ \forall \mathbb{P},$
- $\pi(\phi(s)|x) := \sum_{a\in\mathcal{A}} p_{\mathrm{LLM}}(\phi(s)|x,a)\pi(a|x),\ \forall \pi,$
- $R^\pi(x,\phi(s)) := \int_{s'\in\mathcal{S}} \frac{K(s,s';x,\tau)\pi(s'|x)}{\pi(\phi(s)|x)} R(x,s')ds',\ \forall \pi.$

$K(\cdot)$ is a kernel function, which must satisfy $\int_{s'\in\mathcal{S}} K(s',s;x,\tau) = 1$, and τ is a bandwidth hyperparameter that controls the magnitude of marginalization. The intuition behind DSO is to implicitly augment the data by taking the observations for the neighboring sentences

[2]The estimation target is the *true* PG defined as $\nabla_\theta V(\pi_\theta) = \mathbb{E}_{p(x)\pi_\theta(a|x)p(r,s|x,a)}[\nabla_\theta \log \pi_\theta(a|x) r]$.

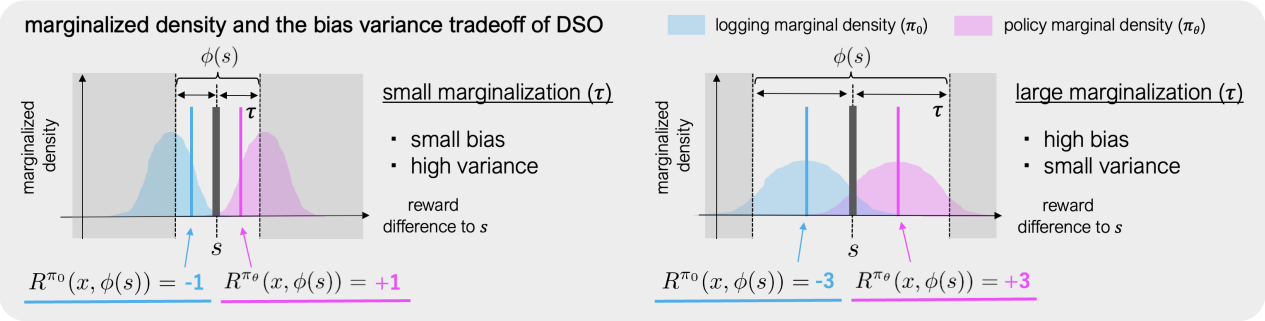


Figure 4: Bias-variance tradeoff of DSO and its relations to the bandwidth hyperparameter (τ) of a kernel function: When τ is large, the overlap between the logging policy (π_0) and the current policy (π_θ) within $\phi(s)$ becomes large, thus the scale of the importance weight becomes small. This contributes to reducing the variance compared to naive IS. In contrast, a small value of τ helps keep the bias small, as the within-neighbor reward shift (i.e., the difference between $R^{\pi_0}(x, \phi(s))$ and $R^{\pi_\theta}(x, \phi(s))$) becomes small. The gray regions are rejected when using a uniform kernel.

into account, as illustrated in Figure 3. Specifically, when using a smooth kernel like a Gaussian kernel, neighboring sentences are weighted proportional to $K(s', s; x, \tau) \propto \exp(-d(s, s'))$, where $d(s, s')$ is the distance between two sentences (e.g., sentence embedding distance). In contrast, when using a piecewise constant kernel, all the sentences within a certain threshold are equally weighted, while all the others are rejected with the weight of 0.

To estimate the policy gradient in the marginalized sentence space induced by kernels, ***Direct Sentence Off-policy Gradient* (DSO)** applies IS as follows.

$$\nabla_\theta V(\pi_\theta) \approx \frac{1}{n} \sum_{i=1}^{n} \underbrace{\frac{\pi_\theta(\phi(s_i)|x_i)}{\pi_0(\phi(s_i)|x_i)}}_{:= w(\phi(s_i), x_i)} \nabla_\theta \log \pi_\theta(\phi(s_i)|x_i)\, r_i.$$

By applying IS on the marginalized sentence space ($\Phi(\mathcal{S})$), DSO avoids large importance weights, making large-scale OPL more scalable regarding the number of candidate prompts, while keeping the bias small by leveraging the similarity among sentences (Section 4.2 provides detailed analyses). While the precise computation of the marginal importance weight ($w(\phi(s), x)$) and the score function ($\nabla_\theta \pi_\theta(\phi(s)|x)$) seems non-trivial, below we present how to train a model to estimate these distributions in a tractable way.

4.1 Estimation of the weighted score function

The key trick of DSO is to use the following expression of the weighted score function:

$$w(\phi(s), x) \nabla_\theta \log \pi_\theta(\phi(s)|x)$$
$$= \mathbb{E}_{(a,s') \sim \pi_\theta(a|x) p_{\mathrm{LLM}}(s'|x,a)} \left[\frac{K(s, s';\, x, \tau) \nabla_\theta \log \pi_\theta(a|x)}{\pi_0(\phi(s)|x)} \right].$$

This expression indicates that DSO can be seen as performing soft rejection sampling on the data (a, s') augmented by π_θ, while correcting the bias in the logged data by applying the inverse propensity of π_0 in the marginalized sentence space. This means that even though we observe only a single prompt in the original logged data, DSO can further distribute the reward observation among multiple prompts that generate similar sentences. This *implicit data augmentation* among multiple counterfactual prompts also contributes to reducing variance.

The above expression of the weighted score function also suggests that our estimation problem of the weighted score function is reduced to only the estimation of $\pi_0(\phi(s)|x)$. This is useful, as $\pi_0(\phi(s)|x)$ does not depend on the parameterized policy (π_θ), and it thus suffices to fit a marginal density model only once before running the policy gradient method. Because the marginal distribution is defined as $\pi_0(\phi(s)|x) = \mathbb{E}_{\pi_0(s'|x)}[K(s, s';\, x, \tau)]$, we can estimate the marginal density via the monte-carlo sampling:

$$\pi_0(\phi(s_i)|x_i) \approx \frac{1}{m} \sum_{j=1}^{m} \mathbb{E}_{s_j \sim \pi_0(s_j|x)}[K(s_i, s_j; x_i, \tau)],$$

where m is the number of the monte-carlo samples. Similarly, we can also estimate the marginal density with function approximation ($f_\psi(x, s) \approx \pi_0(\phi(s)|x)$) using the following loss:

$$\ell(f_\psi) \approx \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}_{(s,s') \sim \pi_0(s|x_i)\pi_0(s'|x_i)} [\\ (f_\psi(x_i, s) - K(s, s';\, x_i, \tau))^2].$$

Since the computation of this loss does not scale with the size of the action space $|\mathcal{A}|$, we can easily apply DSO even when the action (i.e., prompt) space is large or continuous.

4.2 Theoretical Analysis

Here, we analyze the bias and variance of the DSO estimator.

We first introduce a new condition about support in the marginalized sentence space.

Definition 1. *(Similar sentence support) Similar sentence support is satisfied when $\pi_\theta(\phi(s)|x) > 0 \implies \pi_0(\phi(s)|x) > 0$ holds for all $(x, \phi(s)) \in \mathcal{X} \times \Phi(\mathcal{S})$.*

The similar sentence support condition relaxes the action support condition of IS. That is, as $\pi(\phi(s)|x) = \sum_{a \in \mathcal{A}} p_{\mathrm{LLM}}(\phi(s)|a, x)\pi(a|x)$ holds by definition, the similar sentence support condition is always satisfied when the action support condition is satisfied. This

means that deficient support under the similar sentence support is more unlikely happening compared to the action support. Under this condition, we have the following degree of bias of DSO.

Theorem 1. *(Bias of DSO) When the similar sentence support is satisfied, the bias is*

$$\begin{aligned}&\mathrm{Bias}(\widehat{(\nabla_\theta V)}_{\mathrm{DSO}})\\&= \mathbb{E}_{\pi_\theta(\phi(s)|x)}[\nabla_\theta \log \pi_\theta(\phi(s)|x)\Delta_R(\pi_\theta, \pi_0;\, x, \phi(s))]\\&\quad + \mathbb{E}_{\pi_0(\phi(s)|x)\pi_0(s'|x,\phi(s))}[\Delta_{(w\nabla_\theta)}(\phi(s'), \phi(s); x)R(x, s')]\\&\quad + \mathbb{E}_{\pi_\theta(\phi(s)|x)\pi_\theta(s'|x,\phi(s))}[\Delta_{(\nabla_\theta)}(\phi(s), s'; x)R(x, s')].\end{aligned}$$

where $\Delta_R(\pi_\theta, \pi_0;\, x, \phi(s))$ is the difference of $R^\pi(x, \phi(s))$ between π_θ and π_0. $\Delta_{(w\nabla_\theta)}(\phi(s'), \phi(s); x)$ is the difference of weighted score function between $\phi(s')$ and $\phi(s)$. $\Delta_{(\nabla_\theta)}(\phi(s), s'; x)$ is the difference between the score function of $\phi(s)$ and s', which is equivalent to the difference of $\mathbb{E}_{\pi_\theta(a|x,\phi(s))}[\nabla_\theta \log \pi_\theta(a|x)]$ and that of $\mathbb{E}_{\pi_\theta(a|x,s')}[\nabla_\theta \log \pi_\theta(a|x)]$.

Theorem 1 suggests that the bias of DSO comes from three factors. The first term is the dominant term, which arises from the *within-neighbor* reward shift (i.e., the difference between $R^{\pi_0}(x, \phi(s))$ and $R^{\pi_\theta}(x, \phi(s))$), as illustrated in Figure 4. This term becomes small in two cases: (i) when reward does not change too much within $\phi(s)$, and (ii) when the within-neighbor distribution shift of $\pi(s'|x, \phi(s))$ is small. Either case is satisfied when the radius of neighbors (i.e., kernel bandwidth τ) is small. At the same time, smooth kernels like a Gaussian kernel are also useful, as they allocate larger weights to similar sentences depending on the distance from the pivotal sentence. In contrast, the second and third terms are caused by calculating the gradient in the marginalized sentence space ($\Phi(\mathcal{S})$) instead of the original sentence space ($\mathcal{S}$). These terms also become small when the bandwidth hyperparameter τ is small. Thus, a small value of τ is preferable in reducing the bias.

Next, we have the following degree of variance using DSO.

Theorem 2. *(Variance of DSO) When the similar sentence support is satisfied, the conditional variance is expressed as*

$$\begin{aligned}&n\mathbb{V}_{\mathcal{D}|x}(\widehat{(\nabla_\theta V)}_{\mathrm{DSO}})\\&= \mathbb{V}_{\pi_0(s|x)}(w(\phi(s), x)\nabla_\theta \log \pi_\theta(\phi(s)|x)R(x, s))\\&\quad + \mathbb{E}_{p(x)\pi_0(s|x)}[(w(\phi(s), x))^2(\nabla_\theta \log \pi_\theta(\phi(s)|x))^2\sigma_r^2(x, s)].\end{aligned}$$

Compared to the naive (action) IS, the importance weight and the gradient reduce the variance by $\mathbb{E}_{\pi_0(\phi(s)|x)}[\mathbb{V}_{\pi_0(a|x,\phi(s))}(w(a, x))]$ and $\mathbb{E}_{\pi_0(\phi(s)|x)}[\mathbb{V}_{\pi_0(a|x,\phi(s))}(\nabla_\theta \log \pi_\theta(a|x))]$, respectively. $w(x, a)$ is the action importance weight and σ_r is the reward noise.

Theorem 2 suggests that DSO gains variance reduction from two sources: $\nabla_\theta \log \pi_\theta(\phi(s)|x)$ and $w(\phi(s)|x)$. The first variance reduction of $\nabla_\theta \log \pi_\theta(\phi(s)|x)$ comes from the fact that the sentence-based score function is expressed as $\mathbb{E}_{\pi_\theta(a|x,\phi(s))}[\nabla_\theta \log \pi_\theta(a|x)]$, demonstrating the benefit of applying the implicit data augmentation and soft rejection sampling (instead of applying hard rejection sampling). The variance reduction becomes especially large when multiple different prompts result in similar sentences; thus, $\pi_\theta(a|x, \phi(s))$ becomes adequately stochastic. Moreover, by using $w(\phi(s), x)$ instead of $w(a, x)$, we can expect a significant variance reduction as we avoid the variance caused by the within-neighbor importance weight, i.e., $w(a, x; \phi(s)) := \pi_\theta(a|x, \phi(s))/\pi_0(a|x, \phi(s))$. This means that a larger value of τ (i.e., the radius of neighbors) leads to a larger variance reduction.

Summary. Together with the analysis of bias and variance, we can see that the value of τ plays an important role in trading off the bias and variance of DSO, as illustrated in Figure 4. Specifically, when τ is large, the overlap between the logging policy (π_0) and the current policy (π_θ) within a neighbor $\phi(s)$ becomes large, thus the scale of the importance weight becomes small. This contributes to reducing the variance compared to naive IS. In contrast, a small value of τ helps keep the bias small, as the within-neighbor reward shift (i.e., the difference between $R^{\pi_0}(x, \phi(s))$ and $R^{\pi_\theta}(x, \phi(s))$) becomes small. Later in the experiment section, we study how the performance changes with varying values of the bandwidth hyperparameter τ and demonstrate that a smooth kernel like a Gaussian kernel is beneficial for reducing both bias and variance simultaneously.

5 Benchmarks and open-source software

Due to the lack of existing benchmark suites for OPL of prompt policies, we released open-source software called **OfflinePrompts**. This benchmark suite comes with two settings: *synthetic* and *full-LLM* to enable extensive and reproducible experiments. In particular, the full-LLM benchmark simulates movie recommendation tasks with personalized sentence descriptions based on the (sentence-augmented) MovieLens dataset [9]. Moreover, OfflinePrompts also enables prompt tuning on users' own logged data, facilitating the practical application of OPL. The package and documentation are available at https://github.com/aiueola/offline-prompts.

6 Synthetic Experiments

We first evaluate the proposed DSO approach on synthetic benchmarks in OfflinePrompts, since they allow us to explore a wide range of conditions.[3]

6.1 Experiment setting

To generate candidate actions, we first sample 5-dimensional embedding e_a from a normal distribution. Each embedding e_a is a deterministic embedding associated with an action a. Then, to generate logged data, we sample 5-dimensional user and query vectors from a multivariate normal distribution. Next, for each query-action pair (q, a), we sample 5-dimensional sentence embeddings s as

$$s \sim \mathcal{N}(f_s(q, e_a), \sigma_s^2),\ f_s(q, e_a) = c \cdot \sin(q^\top M_q + e_a^\top M_e),$$

where M_q and M_e are coefficient matrices sampled from a uniform distribution. $c = 5.0$ is a scaling factor and $\sigma_s = 1.0$ is the noise level of the action-output mapping. By using the sine function, we simulate a situation where two different prompts (e_a) can result in a similar sentence (s),[4] while preserving the smoothness between the prompt and sentence embedding spaces. Then, a user responds

[3] Our experiment code is available at https://github.com/aiueola/offline-prompts/tree/main/experiments.

[4] For example, prompts containing "romance" and "commedy" may result in similar sentences focusing on the "rom-com" aspect of a movie.

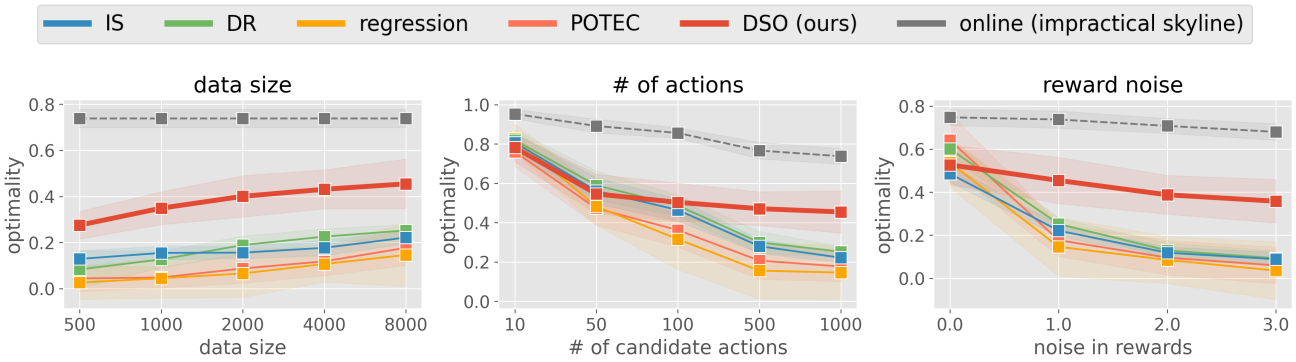


Figure 5: Comparing the performance of the policies learned by various OPL methods with (Left) varying data sizes (n), (Middle) varying number of candidate actions ($|\mathcal{A}|$), and (Right) varying reward noises (σ_r). DSO uses a Gaussian kernel and function approximation of $\pi_0(\phi(s)|x)$. "online" refers to an on-policy trained skyline, which is often infeasible in practice.

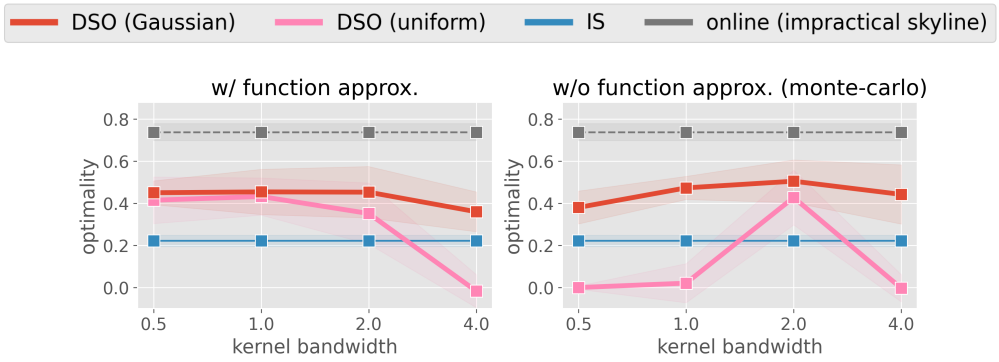


Figure 6: Ablation results of DSO with varying bandwidth hyperparameters (τ), w/ and w/o function approximation of $\pi_0(\phi(s)|x)$, and two kernels, Gaussian and uniform.

to the generated sentence (s) with the following reward function:

$$r \sim \mathcal{N}(f_r(x,s), \sigma_r^2),\ f_r(x,s) = (u^\top M_u + q^\top M_q) M_s s^\top,$$

where M_u, M_q, and M_s are the coefficient matrices and σ_r is the reward noise.

We generate logged data with a softmax logging policy: $\pi_0(a|x) := \exp(\beta_0 \hat{R}_0(x,a))/(\sum_{a\in\mathcal{A}} \exp(\beta_0 \hat{R}_0(x,a)))$. $\hat{R}_0$ is the base reward model, trained on $n_0 = 10000$ of data points collected by the uniform random policy. $\beta_0 = 1.0$ is the inverse temperature.

We compare **DSO** to four baselines: **regression, IS, DR [5], and POTEC [27]**. DR combines IS and regression efficiently for the variance reduction purpose, but its effect is known to be insufficient when the action space is large [25]. POTEC employs a two-stage policy, which first chooses which cluster to use via DR and then chooses action within the cluster via regression. All the baselines estimate the gradient in the action space. For the metrics to compare the OPL methods, we use the *optimality* of the learned policy, defined as $(V(\pi) - V(\pi_{\text{unif}}))/(V(\pi_{\text{opt}}) - V(\pi_{\text{unif}}))$, where π_{opt} is the optimal policy and π_{unif} is the uniform random policy. Note that we also report the performance of the **online** policy, which is trained using on-policy policy gradient (i.e., not using logged data and OPL), as the (impractical) *skyline*.

The experiment varies the following configurations (the **bold** font represents the default value):

(1) **data size**: $n \in \{500, 1000, 2000, 4000, \mathbf{8000}\}$,
(2) **number of candidate actions**: $|\mathcal{A}| \in \{10, 50, 100, 500, \mathbf{1000}\}$,
(3) **reward noises**: $\sigma_r \in \{0.0, \mathbf{1.0}, 2.0, 3.0\}$.

For the ablation of DSO, we additionally report the results with the varying **bandwidth hyperparameters** of $\tau \in \{0.5, \mathbf{1.0}, 2.0, 4.0\}$, {**w/** and w/o} **function approximation** of the marginal density, and two different kernels, {**Gaussian** and uniform}. When not using function approximation, we estimate the marginal density via monte-carlo sampling with $m = 100$ samples. Finally, to evaluate the robustness of DSO to the accuracy of the distance measure in the kernel, we add noise sampled from a normal distribution with std $\Delta_s = 1.0$ to the sentence embeddings. We report the mean and standard deviation of the performance based on 20 random seeds.

6.2 Result

Figure 5 compares the policy learning results of the OPL methods with varying data sizes (n), number of candidate actions ($|\mathcal{A}|$), and reward noises (σ_r), respectively. The results demonstrate that DSO works particularly well in challenging scenarios where the baselines fall short due to variance. Specifically, while we observe a sharp

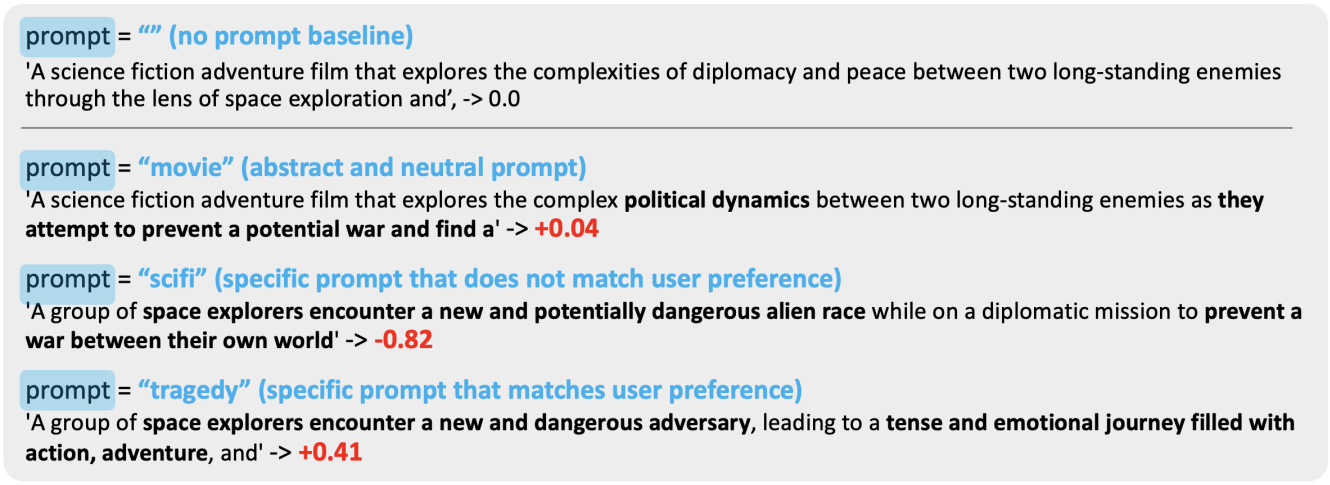


Figure 7: Example of sentences generated with varying prompts and their reward simulated in the full-LLM benchmark. We use the frozen Mistral-7B [11] model to generate descriptions of the movie "Star Trek VI (1991)" with three different prompts, { movie, scifi, tragedy } and highlight sentences that differ from the baseline generated without prompts. The red font indicates the reward simulated by the DistilBert model fine-tuned on the MovieLens dataset. While abstract keywords like "movie" do not make much difference, more specific keywords like "scifi" or "tragedy" can be impactful in the reward simulation.

drop of performance for the baselines when the action space is large ($|\mathcal{A}| \geq 500$) and reward noise is large ($\sigma_r \geq 1.0$), DSO maintains a favorable performance even under these configurations. Moreover, comparing the performance with $|\mathcal{A}| = 1000$ and $\sigma_r = 1.0$, we observe that the performance of DSO at $n = 500$ outperforms that of the baselines at $n = 8000$. This indicates that DSO is far more data-efficient than the baselines when the action space is large, leveraging the similarity among sentences via kernels and performing implicit data augmentation.

Next, we study how the choice of kernels affects the performance of DSO, as shown in Figure 6. The results tell us several interesting findings: using (1) a Gaussian kernel and (2) function approximation improve the robustness of DSO to the choice of bandwidth hyperparameter τ. The first observation is evident from the fact that a Gaussian kernel allocates larger weights to closer sentences compared to a uniform kernel. However, when using monte-carlo estimation, we observe that even a Gaussian kernel needs careful tuning of τ, where a small value of τ incurs high variance and a large value of τ produces non-negligible bias. In contrast, by using function approximation, we can avoid a small value of $\hat{\pi}_0(\phi(s)|x)$, contributing to the variance reduction[5]. Using function approximation thus helps improve the robustness to a small value of τ, and we do not need extensive hyperparameter tuning of τ. This implies that DSO is applicable to practical situations, where a pre-trained model of $\hat{\pi}_0(\phi(s)|x)$ can provide substantial efficiency gains.

7 Full-LLM Experiment with MovieLens

This section compares OPL methods in *a personalized generation task of movie descriptions* using the MovieLens-10M [9] dataset. The MovieLens dataset contains 10M ratings between 71,567 users and 10,681 movies. To use this data in our personalized sentence generation task, we first augment the data by generating a (general) movie description using Mistral-7B [11]. Then, we train a sentence-based reward model on the augmented dataset using DistilBERT [28]. The reward is defined as $10 \times (R(x, s(a)) - R(x, s(\emptyset)))$, where $R(\cdot)$ is the $[0, 1]$-score generated by the DistilBERT sentence discriminator and $s(\emptyset)$ is the sentence generated without adding prompts (referred to as *no-prompt baseline*). We present the simulated rewards for several example prompts in Figure 7.

After obtaining a fine-tuned reward model, we collect the logged data in the following procedure. First, we randomly sample a user (u) and a movie (query) (q) as a context (x). Next, a logging policy (π_0) chooses which prompt (a) to use in the sentence generation task. Then, a frozen LLM generates sentence s, taking the prompt a and query q as the input. Finally, we generate a reward (r) using the reward model.

For the configurations of the full-LLM experiment, we define the logging policy by applying the softmax function on top of the logits learned by the online policy, where we set the inverse temperature hyperparameter to be $\beta_0 = 0.2$. The total number of candidate prompts used in this experiment is $|\mathcal{A}| = 1000$. The data size is $n = 50000$ and the reward noise is $\sigma_r = 0.05$. For DSO, we use a Gaussian kernel with $\tau = 1.0$ to estimate the logging marginal density. The distance between two sentences are measured by the sentence embeddings obtained from the *frozen* Mistral-7B model. We report the results with a total of 25 trials, on 5 different datasets and 5 runs on each data with different random seeds.

Result. Figure 8 compares the performance of the OPL methods by the degree of improvement that the learned policy observed over the sentences generated without prompts (which we call *no-prompt* baseline). The results indicate that DSO often improves

[5]This is because, for example, when the true marginal density is 1e-5, estimating it as 1e-5 and 1e-4 does not change the MSE loss too much. However, in terms of variance, 1e-4 and 1e-5 make a significant difference. Using function approximation, we can avoid being too precise about small values of the marginal density.

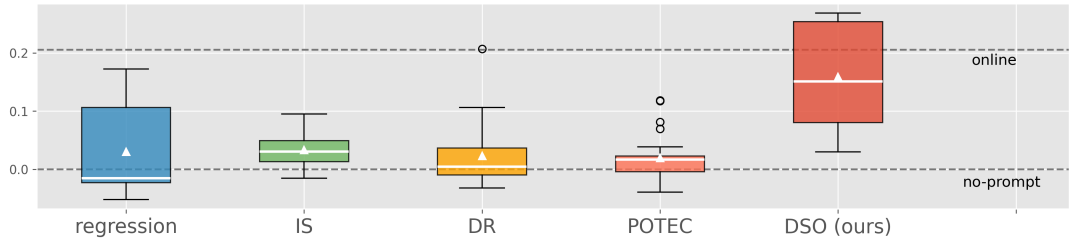


Figure 8: Performance comparison of OPL methods in the full-LLM experiment.: The policy value indicates how much improvement of reward we have by using a (learned) prompt policy compared to the sentence generation without prompts (called *no-prompt baseline*). From the top, the horizontal lines refer to the value of an on-policy trained (impractical) skyline and the no-prompt baseline.

the effectiveness of the sentences more than other OPL methods, by effectively leveraging the information about similar sentences. Specifically, DSO is more resilient to performance corruption than IS by substantially reducing the variance and than regression by reducing the bias. As a result, we have **5x increase of the performance than other baselines on average**. It should also be worth noting that this result is observed for the off-the-shelf embeddings of sentences, which do not require extensive tuning of the embedding model. This minimizes the difficulty in applying the proposed OPL method in practice. However, learning (application-specific) embeddings that further improve the performance of DSO is an interesting direction for future work.

8 Conclusion and Future Work

This paper studied how to use naturally logged user feedback to optimize a prompt policy for language generation. We started by formulating the problem as OPL of contextual bandits with auxiliary outputs. Then, we pointed out the limitations of the naive approaches – (1) existing OPL methods often suffer from the large action space of prompts and (2) even though we observe generated sentences, existing methods do not use the information about these sentences. To overcome these shortfalls, we proposed *Direct Setence Off-policy gradient* (DSO), which applies importance sampling taking the similarity of sentences into account. We also show the effectiveness of the proposed approach in both theoretical and empirical ways. Furthermore, our benchmarks suite called OfflinePrompts, provided as open-source software, accelerates future research and practical application of prompt-guided sentence generation from logged data.

As a remark, deriving a DR-style variant, which introduces a control variate for further variance reduction [5], is non-trivial for DSO. Studying a way to efficiently combine kernel IS and regression in the DSO framework would be a promising future direction. Additionally, a good representation or distance measure of sentences that further improves the performance of DSO would also be worth exploring. Finally, applying a similar idea to other generative AI applications, such as text-to-image diffusion models [23], and extending the benchmark by publishing relevant real-world data can be interesting future works.

Acknowledgments

This research was supported in part by NSF Awards IIS-2312865 and OAC-2311521. Haruka Kiyohara and Yuta Saito are supported by the Funai Overseas Scholarship.

Additionally, we thank Kiante Brantley and Aaron Tucker for their advice on the use of LLMs and large computational resources, and Zhaolin Gao for the discussion on the sentence-based reward simulator in the full-LLM benchmark. We also thank the anonymous reviewers for the valuable discussion to improve the manuscripts.

References

[1] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional AI: Harmlessness from AI Feedback. *arXiv preprint arXiv:2212.08073* (2022).

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems* 33 (2020), 1877–1901.

[3] 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 (2017).

[4] Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric Xing, and Zhiting Hu. 2022. RLPrompt: Optimizing Discrete Text Prompts with Reinforcement Learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 3369–3391.

[5] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly Robust Policy Evaluation and Learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*. 1097–1104.

[6] Vikranth Dwaracherla, Seyed Mohammad Asghari, Botao Hao, and Benjamin Van Roy. 2024. Efficient Exploration for LLMs. In *Proceedings of the 41th International Conference on Machine Learning*, Vol. 235. 12215–12227.

[7] Justin Fu, Mohammad Norouzi, Ofir Nachum, George Tucker, Alexander Novikov, Mengjiao Yang, Michael R Zhang, Yutian Chen, Aviral Kumar, Cosmin Paduraru, et al. 2020. Benchmarks for Deep Off-Policy Evaluation. In *International Conference on Learning Representations*.

[8] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline A/B Testing for Recommender Systems. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. 198–206.

[9] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[10] 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*. PMLR, 1645–1654.

[11] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel,

Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

[12] Nathan Kallus and Angela Zhou. 2018. Policy Evaluation and Optimization with Continuous Treatments. In *International Conference on Artificial Intelligence and Statistics*. 1243–1251.

[13] Haruka Kiyohara, Ren Kishimoto, Kosuke Kawakami, Ken Kobayashi, Kazuhide Nakata, and Yuta Saito. 2024. Towards Assessing and Benchmarking Risk-Return Tradeoff of Off-Policy Evaluation. *International Conference on Learning Representations* (2024).

[14] Haruka Kiyohara, Yuta Saito, Tatsuya Matsuhiro, Yusuke Narita, Nobuyuki Shimizu, and Yasuo Yamamoto. 2022. Doubly Robust Off-Policy Evaluation for Ranking Policies under the Cascade Behavior Model. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining*. 487–497.

[15] Vijay Konda and John Tsitsiklis. 1999. Actor-Critic Algorithms. *Advances in Neural Information Processing Systems* 12 (1999).

[16] Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. 2023. Rlaif: Scaling Reinforcement Learning from Human Feedback with AI Feedback. *arXiv preprint arXiv:2309.00267* (2023).

[17] Xiaoqiang Lin, Zhongxiang Dai, Arun Verma, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. 2024. Prompt Optimization with Human Feedback. *arXiv preprint arXiv:2405.17346* (2024).

[18] Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. 2021. Deployment-Efficient Reinforcement Learning via Model-Based Offline Optimization. In *International Conference on Learning Representations*.

[19] 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 (2022), 27730–27744.

[20] Doina Precup, Richard S. Sutton, and Satinder P. Singh. 2000. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the 17th International Conference on Machine Learning*. 759–766.

[21] Noveen Sachdeva, Yi Su, and Thorsten Joachims. 2020. Off-Policy Bandits with Deficient Support. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 965–975.

[22] Noveen Sachdeva, Lequn Wang, Dawen Liang, Nathan Kallus, and Julian McAuley. 2024. Off-policy evaluation for large action spaces via policy convolution. In *Proceedings of the ACM on Web Conference 2024*. 3576–3585.

[23] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems* 35 (2022), 36479–36494.

[24] Yuta Saito, Shunsuke Aihara, Megumi Matsutani, and Yusuke Narita. 2021. Open Bandit Dataset and Pipeline: Towards Realistic and Reproducible Off-Policy Evaluation. In *35th Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

[25] Yuta Saito and Thorsten Joachims. 2022. Off-policy evaluation for large action spaces via embeddings. In *Proceedings of the 39th International; Conference of Machine Learning*, Vol. 162. 19089–19122.

[26] Yuta Saito, Qingyang Ren, and Thorsten Joachims. 2023. Off-Policy Evaluation for Large Action Spaces via Conjunct Effect Modeling. In *Proceedings of the 40th International; Conference of Machine Learning*, Vol. 202. 29734–29759.

[27] Yuta Saito, Jihan Yao, and Thorsten Joachims. 2024. POTEC: Off-Policy Learning for Large Action Spaces via Two-Stage Policy Decomposition. *arXiv preprint arXiv:2402.06151* (2024).

[28] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[29] Charlie Victor Snell, Ilya Kostrikov, Yi Su, Sherry Yang, and Sergey Levine. 2022. Offline RL for Natural Language Generation with Implicit Language Q Learning. In *The 11th International Conference on Learning Representations*.

[30] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to Summarize with Human Feedback. *Advances in Neural Information Processing Systems* 33 (2020), 3008–3021.

[31] Adith Swaminathan and Thorsten Joachims. 2015. Batch Learning from Logged Bandit Feedback through Counterfactual Risk Minimization. *The Journal of Machine Learning Research* 16, 1 (2015), 1731–1755.