Diffusion for Offline Bandit

Xinyue Ying **Shouchen Zhou**
School of Information Science and Technology
ShanghaiTech University
{yingxy2024, zhoushch}@shanghaitech.edu.cn

In this project, we aim to improve offline bandit learning by combining policy gradient methods with discrete diffusion models for data augmentation. For stochastic bandits, we pretrain arm-specific policies using offline trajectories, but limited coverage and bias in the data can hinder performance. To address this, we propose using discrete diffusion models to generate additional trajectories and enhance policy initialization.

For contextual bandits, where high-dimensional contexts make reward estimation unreliable, we use conditional diffusion models to model $p(r \mid x, a)$ and estimate both expected reward and uncertainty via sampling. Actions are then selected using an uncertainty-aware scoring rule. Overall, our approach leverages diffusion-based generative modeling to enhance offline learning in both stochastic and contextual bandit settings.

1 Stochastic Bandit

1.1 Motivation

In the settings of the stochastic bandit, which are traditionally addressed using incremental or gradient-based methods. In the offline setting, policy gradient techniques offer a principled way to learn a parametric policy from the collected trajectory offline dataset.

We could parameterize the policy over arms using a softmax distribution:

$$\pi_\theta(a) = \frac{e^{\beta_t H_\theta(a)}}{\sum_{a'} e^{\beta_t H_\theta(a')}}$$

where $H_\theta(a)$ denotes a learnable preference function for arm a. The objective is to maximize the expected cumulative reward over trajectories, each of which could be formulated as

$$\psi = (a_0, r_1, \cdots, , a_{T-1}, r_T)$$

Each trajectory ψ is a sample of the random variable $\Psi \sim P_\theta$, with a probability of occurrence of

$$P_\theta(\psi) = \prod_{t=0}^{T-1} \pi_\theta(a_t)$$

Thus, the objective function is

$$J(\theta) = \mathbb{E}_{\Psi \sim P_\theta}\left[R(\Psi)\right] = \sum_{\psi} P_\theta(\psi) R(\psi)$$

This can be optimized via gradient ascent on the offline dataset $\mathcal{D}$ to produce a warm-start policy for each arm. The gradient could be computed as

$$\nabla_\theta J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T-1} \left[\left(G_t^i - b\right) \beta_t \left(\nabla_\theta H_\theta\left(a_t^i\right) - \sum_a \pi_\theta(a) \nabla_\theta H_\theta(a) \right) \right]$$

This policy pretraining strategy can alleviate cold-start issues and serve as a strong initialization for downstream learning. However, directly optimizing on $\mathcal{D}$ may have the following challenges:

Proposal for SI252 Reinforcement Learning Final Project.

1. Limited coverage: Offline data may be biased toward a subset of arms, leading to poor generalization.
2. Reward bias: The observed rewards may reflect the behavior of an unknown logging policy, introducing selection bias.
3. Poor generalization: Policies may overfit to observed trajectories and fail to explore high-value alternatives.

1.2 Method: Trajectory Augmentation via Discrete Diffusion Models

To address these limitations, we propose to incorporate a discrete diffusion model to model and generate synthetic trajectories. This model learns the underlying generative structure of arm-reward sequences from $\mathcal{D}$, enabling us to augment the dataset with diverse and potentially high-reward samples before policy learning.

We aim to obtain more robust and better-initialized arm preferences by combining trajectory generation with policy gradient pretraining. Our proposed framework consists of the following steps:

1. Train a discrete diffusion model on the original offline dataset to generate augmented trajectories;
2. Combine the original and synthetic data to pretrain arm-specific policies using a policy gradient objective;
3. Deploy the pretrained policy for downstream tasks or further online fine-tuning.

This approach integrates the generative modeling capabilities of diffusion models with the direct optimization benefits of policy gradient methods, providing a novel solution to the warm-up for the offline bandit learning problem.

2 Contextual Bandit

2.1 Motivation

In offline contextual bandits, the goal is to learn a policy $\pi(a \mid x)$ that maps context $x \in \mathbb{R}^{d_x}$ to actions $a \in \mathcal{A}$ (discrete and low-dimensional) using logged data (x, a, r). The goal is to maximize the expected reward:

$$\mathbb{E}_{x\sim\mathcal{D}_x, a\sim\pi(\cdot|x)}[r \mid x, a]$$

In practice, this setup is challenged by **High-dimensional, sparse contexts** (user behavior embeddings). Each specific context x may appear only once or a few times in the offline data. As a result, empirical reward statistics (mean$(r \mid x, a)$) are unreliable.

Traditional approaches using mean regression $\hat{\mu}(x, a)$ (neural bandit or LinUCB) often fail under these conditions due to data sparsity and poor generalization. These regressors output point estimates $(\hat{\mu})$ and do not characterize the credibility of the estimates at all.

2.2 Method

We propose using a **conditional diffusion model** to model the reward distribution given (x, a): $p_\phi(r \mid x, a)$

This model serves as a generative regressor: by sampling $r^{(1)}, \ldots, r^{(N)} \sim p_\phi(\cdot \mid x, a)$, we obtain:

$$\hat{\mu}(x, a) = \frac{1}{N}\sum_{i=1}^{N} r^{(i)}, \quad \hat{\sigma}^2(x, a) = \frac{1}{N}\sum_{i=1}^{N}(r^{(i)} - \hat{\mu})^2$$

Model Architecture:

- Input: concatenation of $[f(x), a, r_t, t]$, where $f(x)$ is a context encoder (MLP)
- Network: 4-layer MLP with SiLU activations.

- Loss: Denoising Score Matching:

$$\mathcal{L} = \mathbb{E}_{x,a,r,t,\varepsilon} \left\| \varepsilon_\phi(x, a, r_t, t) - \varepsilon \right\|_2^2$$

Inference For each candidate action a, run N stochastic reverse diffusion steps to obtain reward samples and compute $\hat{\mu}$ and $\hat{\sigma}$. Select the action via:

$$a^\star = \arg\max_a \left[\hat{\mu}(x, a) + \lambda \cdot \hat{\sigma}(x, a) \right]$$

2.3 Algorithm

Step 1: Offline Training

Train diffusion on (x, a, r) tuples with standard DDPM denoising loss.

Step 2: Online Arm Selection

1. For each arm a, run N reward samples via diffusion.
2. Compute $\hat{\mu}(x, a), \hat{\sigma}(x, a)$.
3. Score: $\hat{\mu}(x, a) + \lambda \cdot \hat{\sigma}(x, a)$.
4. Select arm with highest score.
5. After a certain number of rounds, re-input the data collected online into the diffusion training.

2.4 Experiment

Dataset: MovieLens-20M.
Baselines:

- MLP mean regression $\hat{\mu}(x, a)$
- Gaussian regression
- Diffusion reward model (ours)