条件生成模型是决策所需的全部吗？

Anurag Ajay[*†§¶], **Yilun Du** [*§¶], **Abhi** [Gupta*‡§¶], **Joshua Tenenbaum**[(¶)], **Tommi Jaakkola**[(‡§¶)], **Pulkit Agrawal**[†§¶]
Improbable AI Lab[†]Operations
Research Center[‡]
计算机科学与人工智能实验室[§]麻省理工学院[¶]

内容摘要

条件生成建模技术的最新进展使得仅凭语言描述就能生成高质量图像成为可能。我们研究了这些方法能否直接解决顺序决策问题。我们不是从强化学习（RL）的角度来看待决策问题，而是从条件生成建模的角度来看待决策问题。令我们惊喜的是，我们发现我们所提出的策略在标准基准上优于现有的离线 RL 方法。通过将策略建模为回报条件扩散模型，我们说明了如何规避动态编程的需要，从而消除传统离线 RL 所带来的许多复杂性。通过考虑其他两个条件变量：约束和技能，我们进一步证明了将政策建模为条件扩散模型的优势。在训练过程中对单个约束条件或技能设定条件，会导致测试时的行为可以同时满足多个约束条件或展示技能组合。我们的研究结果表明，条件生成模型是一种强大的决策工具。

1 简介

在过去几年中，条件生成模型在一系列领域取得了令人瞩目的成果，包括根据文本描述生成高分辨率图像（DALL-E、ImageGen）（Ramesh等人，2022年；Saharia等人，2022年）、语言生成（GPT）（Brown等，人2020年年）以及数学问题的逐步解答（Minerva）（Lewkowycz等人，2022）。生成模型在无数领域的成功促使我们将其应用于决策。

方便的是，目前已有大量关于从已运行系统记录的数据中恢复高性能策略的研究（Kostrikov 等人，2022 年；Kumar 等人，2020 年；Walke 等人，2022 年）。在现实世界中，与环境互动并不总是可能的，探索性决策可能会带来致命后果（Dulac-Arnold 等人，2021 年）。有了这种离线数据集，决策问题就可以简化为学习轨迹的概率模型，在这种情况下，生成模型已经取得了成功。

在离线决策中，我们的目标是通过拼接训练数据集中的次优奖励标注轨迹来恢复最优奖励最大化轨迹。之前的研究（Kumar 等人，2020 年；Kostrikov 等人，2022 年；Wu 等人，2019 年；Kostrikov 等人，2021 年；Dadashi 等等等人，2021 年；Ajay 人，2020 年；Ghosh 人，2022 年）通过强化学习（RL）解决了这一问题，该方法使用动态编程进行轨迹拼接。为了实现动态编程，这些研究学习了一个*价值*函数，用于估算给定状态下的奖励折现总和。然而，由于函数近似、偏离策略学习和自举（bootstrapping）等原因，价值函数估计容易出现不稳定性，这*三者*合称为*致命三要素*（Sutton & Barto，2018）。此外，为了稳定离线状态下的价值估计，这些研究依赖启发式方法将策略保持在数据集分布范围内。这些挑战使得现有离线 RL 算法难以扩展。

*表示贡献相同。对应，aajay@mit.eduyilundu@mit.edu、 abhig@mit.edu

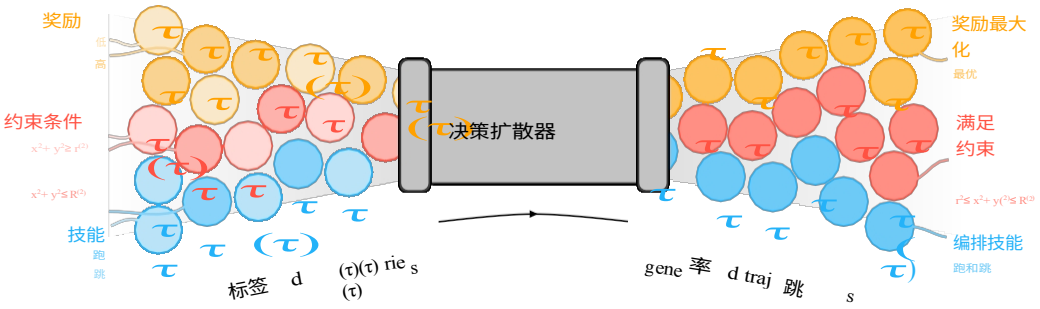


图 1：**利用条件生成模型进行决策。**将决策制定作为一个条件生成建模问题，可使我们最大限度地提高回报、满足约束条件并组合技能。

在本文中，我们提出了一个问题：能否在不依赖数值估计的情况下，通过动态编程来拼接次优轨迹，从而获得最优轨迹？由于条件扩散生成模型可以通过组合训练数据生成新的数据点（Saharia 等人，2022 年；Ramesh 等人，2022 年），因此我们在离线决策中将其用于轨迹拼接。给定一个固定的有回报标签的轨迹数据集，我们调整扩散模型（Sohl-Dickstein 等人，2015 年）来学习轨迹的回报条件模型。在推理过程中，我们使用*无分类器引导*的*低温采样*，我们假设这种方法可以隐式地执行动力学编程，以捕捉数据集中的最佳行为，并收集回报最大化轨迹（详见附录 A）。在标准的 D4RL 任务中，我们的直接条件生成建模公式优于现有方法（Fu 等人，2020 年）。

通过条件生成模型的视角来观察离线决策，可以超越收益条件（图 1）。请看一个例子（详见附录 A）：一个具有线性动力学特性的机器人在一个包含两个同心圆的环境中航行（图 2）。我们给定了机器人的状态-动作轨迹数据集，每个数据集都满足两个约束条件中的一个：(i) 机器人的最终位置在大圆内，(ii) 机器人的最终位置在小圆外。通过条件扩散建模，我们可以利用数据集来学习一个约束条件模型，该模型可以生成满足任意一组约束条件的轨迹。在推理过程中，学习到的轨迹模型可以合并数据集中的约束条件，生成满足组合约束条件的轨迹。图 2 显示，约束条件模型可以生成轨迹，使机器人的最终位置位于同心圆之间。

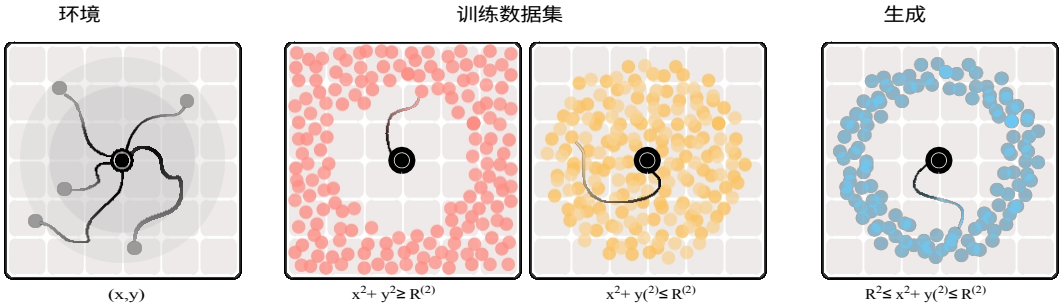


图 2：**示例。**我们展示了 2d 机器人导航环境，以及从环境中提取的数据集中的轨迹所满足的约束条件。我们展示了条件扩散模型生成满足组合约束条件的轨迹的能力。

在这里，我们展示了将策略建模为条件生成模型的好处。首先，以约束条件为条件，政策不仅能生成满足单个约束条件的行为，还能在测试时通过灵活组合约束条件生成新的行为。此外，技能条件允许策略不仅模仿单个技能，还能通过组合这些技能产生新的行为。我们通过基于状态序列的扩散概率模型（Ho 等人，2020 年）实现了这一想法，该模型被称为*决策扩散器*（*Decision Diffuser*），如图 1 所示。总之，我们的贡献包括：**(i)** 说明条件生成模型是离线决策制定的有效工具；**(ii)** 使用无分类器指导的低温采样，而不是动态编程；**(iii)** 使用无分类器指导的低温采样，而不是动态编程。

(iii) 利用 条件生成建模框架，在推理过程中灵活地结合约束条件和组合技能。

2 背景知识

2.1 强化学习

我们将顺序决策问题表述为贴现马尔可夫决策过程（MDP），由元组 $(\rho_0, S, A, T, R, \gamma)$ 定义，其中 ρ_0 是初始状态分布，S 和 A 是状态空间和行动空间，$T : S\times A\to S$ 是转换函数，$R : S\times A\times S\to \mathbb{R}$ 给出任何转换时的奖励，$\gamma\in[0, 1)$ 是贴现因子（Puterman，2014）。代理人的行为是随机策略 $\pi : S \to\Delta_A$，产生一系列状态-Σtion-回报转换或轨迹 $\tau := (s_k, a_k, r_{(k)})_{k\geq 0}$ 概率 $p_\pi(\tau)$ 并返回 $R(\tau) := \sum_{k\geq 0}\gamma^{(k)}r^{k}_{)}$。RL 的标准目标是找到回报最大的策略 $\pi^* = \arg\max_{(\pi)}\mathbb{E}_{(\tau)\sim(p)\,(\pi)}[R(\tau)]$。

时差学习 TD 方法（Fujimoto 等人，2018 年；Lillicrap 等人，2015 年）估计 $Q^*(s, a) = \mathbb{E}_{\tau\sim p^*}[R(\tau)\, s_0 = s, a_0 = a]$，即从状态 s 开始并采取 a 行动时，在最优策略 π^* 下取得的收益，并使用参数化的 *Q 函数*。这需要最小化以下 TD 损失：

$$L_{(\mathrm{TD})}(\vartheta) = \mathbb{E}_{((s,a,r,s')\sim(\mathcal{D}))}[(r+\gamma\max_{a'\in A} Q_{(\bar{\vartheta})}(s', a') - Q_{(\vartheta)}(s, a))^{(2)}] \quad (1)$$

连续行动空间还要求学习参数策略 $\pi_{(\phi)}(a\, s)$，它在方程 1 中扮演最大化行动的角色。这就产生了一个必须最大化的政策目标：

$$J(\phi) = \mathbb{E}_{(s)\sim(\mathcal{D})\,(a)\sim(\pi)\,(\phi)}[Q(s, a)] \quad (2)$$

在这里，过渡数据集 $\mathcal{D}$ 随着代理与环境的交互而演变，Q_ϑ 和 π_ϕ 是一起训练的。这些方法利用了函数逼近、非政策学习和引导，在实践中导致了一些不稳定性（Sutton，1988；Van Hasselt 等人，2018）。

离线 RL 在这种情况下，我们必须从未知行为策略 μ 收集的固定转换数据集中找到回报最大化策略（Levine 等人，2020 年）。使用 TD-learning 会导致状态访问分布 $d^{(\pi)}(s)$ 偏离数据集分布 $d^{(\mu)}(s)$。反过来，策略 π_j 开始采取与数据中已出现的行动大相径庭的行动。离线 RL 算法通过在 TD 学习过程中直接施加 $D(d^{(\pi)}\| d^{(\mu)})$ 形式的约束（其中 D 是某种发散度量）来解决这种分布偏移问题。现在的约束优化问题需要额外的超参数调整和实施启发式方法才能达到合理的性能（Kumar 等人，2021 年）。相比之下，决策扩散器没有这些缺点。它不需要估计任何类型的 *Q 函数*，从而完全避开了 TD 方法。它也不会面临分布偏移的风险，因为生成模型是通过最大似然估计来训练的。

2.2 扩散概率模型

扩散模型（Sohl-Dickstein 等人，2015 年；Ho 等人，2020 年）是一种特定类型的生成模型，可从数据集中学习数据分布 $q(x) := \{x^i\}_{0\leq i<M}$。这些模型主要用于根据文本描述合成高质量图像（Saharia 等人，2022 年；Nichol 等人，2021 年）。在此，数据生成过程采用预定义的前向噪声过程建模 $q(x_{(k)(+1)}\,|\,x_{(k)}) := \mathcal{N}(x_{(k)(+1)};\sqrt{\alpha_k}x_{(k)}, (1-\alpha_{(k)})\boldsymbol{I})$ 和一个可训练的反向过程 $p_\vartheta(x_{k-1}|x_{(k)}) := \mathcal{N}(x_{k-(}\in\mu_{(\vartheta)}(x_k, k), \Sigma_{(k)})$，其中 (μ, Σ) 表示均值为 μ 且方差为 Σ 的高斯分布，$\alpha_{(k)}$ 决定方差计划，$x_0 := x$ 为样本，$x_{(1)}, x_2, \ldots, x_{K-1}$ 为潜变量，$x_{(K)}$ （从高斯噪声开始，通过一系列 "去噪 "步骤迭代生成样本。

尽管可以优化 $\log p_\vartheta$ 的可变下限来训练扩散模型，但 Ho 等人（2020 年）提出了一种简化的替代损失：

$$L_{\mathrm{denoise}}(\vartheta) := \mathbb{E}_{k\sim[1,K],x\sim q,\epsilon\sim\mathcal{N}(0,\boldsymbol{I})}[\|\epsilon - \epsilon_\vartheta(x_k, k)\|^2]。 \quad (3)$$

用深度神经网络参数化的预测噪声 $\epsilon_{(\vartheta)}(x_k, k)$ 估计了加入数据集样本 $x(0)$ 以产生噪声 $x_{(k)}$ 的噪声 $\epsilon\sim\mathcal{N}(0, I)$。这相当于预测 $p_{(\vartheta)}(x_{k-1}\,|\,x_{(k)})$ 的均值，因为 $\mu_{(\vartheta)}(x_{(k)}, k)$ 可以作为 $\epsilon_{(\vartheta)}(x_{(k)}, k)$ 的函数来计算（Ho 等人，2020）。

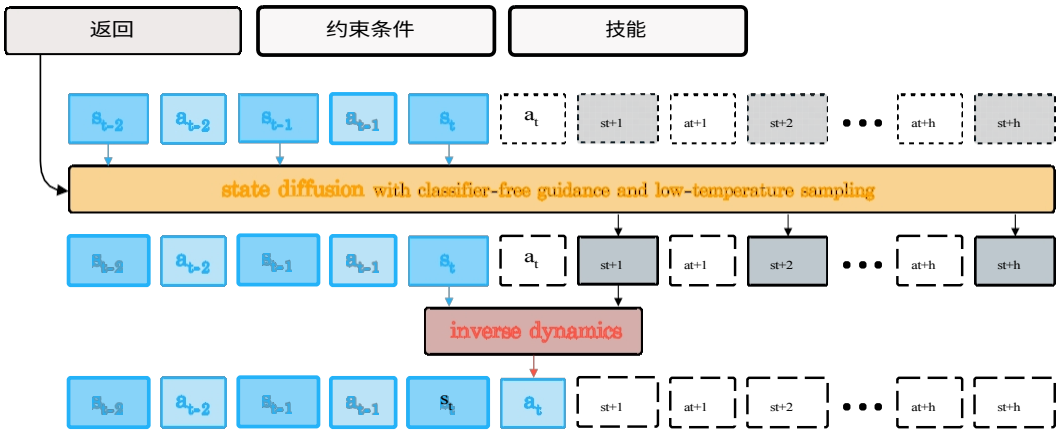


图 3：**使用决策扩散器进行规划。**给定当前状态 s_t 和条件，决策扩散器使用无分类器引导和低温采样来生成一系列未来状态。然后，决策扩散器利用反动力学提取并执行行动 a_t，以实现当前的未来状态 s_{t+1}。

扩散模型与分数匹配之间的等价性（Song 等，2021 年），表明 $\epsilon_\theta(x_k, k) \propto \nabla_{x_k} \log p(x_k)$，从而产生了两种调节方法：分类器引导法（Nichol & Dhariwal，2021）和无分类器法（Ho & Salimans，2022）。前者需要在噪声数据上训练一个额外的分类器 $p_\phi(y \mid x_k)$，使样本测试时可能会产生扰动噪声 $\tilde{\epsilon}_\theta(x_k, k) = \epsilon_\theta(x_k, k) - \omega\sqrt{1-\bar{\alpha}_k}\nabla_{x_k}\log p_\phi(y \mid x_k)$，其中 ω 被称为引导尺度。后者并不单独训练分类器，而是修改原始训练设置，以学习噪声的条件 $\epsilon_\theta(x_k, y, k)$ 模型和无条件 $\epsilon_\theta(x_k, k)$ 模型。实际上，无条件噪声表示为条件噪声 $\epsilon_\theta(x_k, \varnothing, k)$，其中假值 $\varnothing$ 代替 y。扰动噪声 $\epsilon_\theta(x_k, k) + \omega(\epsilon_\theta(x_k, y, k) - \epsilon_\theta(x_k, k))$ 用于随后生成样本。

3 利用决策扩散器建立生成模型

利用离线数据解决 RL 问题非常有用，既不需要依赖 TD 学习，也不会有分布偏移的风险。为此，我们将顺序决策表述为条件生成建模的标准问题：

$$\max_\theta \mathbb{E}_{\tau \sim \mathcal{D}}[\log p_\theta(\boldsymbol{x}_0(\tau) \mid \boldsymbol{y}(\tau))] \quad (4)$$

我们的目标是用 p_θ 来估计条件数据分布，这样以后就可以根据有关信息 $y(\tau)$ 生成轨迹 $\boldsymbol{x}_0(\tau)$ 的部分内容。y 的例子可以包括轨迹下的收益、轨迹满足的约束条件或轨迹中展示的技能。我们根据条件扩散过程构建生成模型：

$$q(\boldsymbol{x}_{k+1}(\tau) \mid \boldsymbol{x}_k(\tau)), \quad p_\theta(\boldsymbol{x}_{k-1}(\tau) \mid \boldsymbol{x}_k(\tau), y(\tau)) \quad (5)$$

通常，q 代表正向去噪过程，而 p_θ 代表反向去噪过程。下面，我们将讨论如何利用扩散进行决策。首先，我们将在第 3.1 节中讨论扩散的建模选择。接下来，我们将在第 3.2 节中讨论如何利用无分类器引导来捕捉轨迹的最佳方面。然后，我们将在第 3.3 节讨论条件扩散模型可能实现的不同行为。最后，我们将在第 3.4 节讨论我们的方法的实际 训练细节。

3.1 状态扩散

在图像中，扩散过程适用于图像中的所有像素值。因此，应用类似的过程对轨迹的状态和行动进行建模是很自然的。然而，在强化学习环境中，直接使用扩散过程对行动建模有以下几个问题

实际问题。首先，在 RL 中，状态通常是连续的，而动作则更加多样，通常是离散的。此外，通常以关节扭矩表示的动作序列往往频率较高，不够平滑，因此更难预测和建模（Tedrake，2022 年）。鉴于这些实际问题，我们选择只对状态进行扩散，定义如下：

$$\boldsymbol{x}_k(\tau) := (s_t, s_{t+1}, \ldots, s_{t+H-1})_k \tag{6}$$

这里，k 表示前向过程中的时间步长，t 表示轨迹 τ 中某个状态被访问的时间。我们将 $\boldsymbol{x}_{(k)}(\tau)$ 表示为一个二维数组，序列的每个时间步为一列。

反动力学作用从扩散模型中采样状态不足以定义控制器。然而，在 $\boldsymbol{x}_{(0)}(\tau)$中的任意时间步 t，通过估计导致状态 s_t变为 $s_{(t)(+1)}$的动作 $a(t)$ 可以推断出一种策略。给定两个连续状态，我们根据逆动力学模型生成一个动作（Agrawal 等人，2016；Pathak 等人，2018）：

$$a_t := f_{(\phi)}(s_t, s_{(t)(+1)}) \tag{7}$$

请注意，用于训练反向过程 $p_{(\vartheta)}$的离线数据也可以用来学习 f。我们在表 2 中说明了直接扩散状态分布的设计选择，以及利用反向动力学模型预测行动的设计选择，是如何比同时扩散状态和行动显著提高性能的。此外，我们还在附录 F 中对何时使用反向 动力学和何时对行动进行扩散进行了实证比较和分析。

3.2 使用无分类指导进行规划

有了代表数据集中不同轨迹的扩散模型，接下来我们将讨论如何利用扩散模型进行规划。要将模型用于规划，有必要对特征 $\boldsymbol{y}(\tau)$ 的扩散过程附加条件。一种方法是训练分类器 $p_{(\phi)}(\boldsymbol{y}(\tau) | \boldsymbol{x}_{(k)}(\tau))$ 从噪声轨迹 $\boldsymbol{x}_{(k)}(\tau)$ 中预测 $\boldsymbol{y}(\tau)$。如果 $\boldsymbol{y}(\tau)$ 代表轨迹下的收益，则需要估计 Q *函数*，这就需要一个单独的、复杂的动态编程过程。

避免动态编程的一种方法是直接根据离线数据集中的收益 $\boldsymbol{y}(\tau)$ 训练条件扩散模型。然而，由于我们的数据集由一组次优轨迹组成，条件扩散模型将受到这些次优行为的污染。为了规避这个问题，我们利用无分类器引导（Ho & Salimans，2022 年）和低温采样来提取数据集中的高可能性轨迹。我们发现，这些轨迹与数据集中的最佳行为集相对应。关于 Q 函数引导和无分类器引导的详细讨论，请参阅附录 K。从形式上看，要实现无分类器引导，需要从高斯噪声 $\boldsymbol{x}_{(K)}(\tau)$开始对 $\boldsymbol{x}_{(0)}(\tau)$进行采样，并在每个中间时间步用扰动噪声将 $\boldsymbol{x}_{(k)}(\tau)$细化为 $\boldsymbol{x}_{k-(1)}(\tau)$：

$$\epsilon\hat{} := \epsilon_{(\vartheta)}(\boldsymbol{x}_{(k)}(\tau), \varnothing, k) + \omega(\epsilon_{(\vartheta)}(\boldsymbol{x}_{(k)}(\tau), \boldsymbol{y}(\tau), k) - \epsilon_{(\vartheta)}(\boldsymbol{x}_{(k)}(\tau), \varnothing, k)), \tag{8}$$

其中，应用于 $(\epsilon_{(\vartheta)}(\boldsymbol{x}_{(k)}(\tau), \boldsymbol{y}(\tau), k)$ $\epsilon_{(\vartheta)}(\boldsymbol{x}_{(k)}(\tau), \varnothing, k))$ 的标量 ω 试图增强和提取数据集中显示 $\boldsymbol{y}(\tau)$ *的*轨迹的最佳部分。有了这些要素，从决策扩散器中采样就变得类似于 RL 中的规划。首先，我们观察环境中的状态。接下来，我们利用以 $\boldsymbol{y}$ 和观察到的最后 C *个*状态的历史记录为条件的扩散过程，对水平线以后的状态进行采样。最后，我们利用逆动力学模型确定应采取的行动，以达到最直接的预测状态。这一过程在算法 1 中描述的标准 后退视界控制环中重复进行，如图 3 所示。

3.3 收益以外的条件变量

到目前为止，我们还没有明确定义条件变量 $\boldsymbol{y}(\tau)$。虽然我们提到它可以是轨迹下的收益，但我们也可以考虑引导我们的扩散过程走向满足相关约束条件或展示特定行为的状态序列。

收益最大化 为了生成收益最大化的轨迹，我们将噪声模型设定为轨迹收益的条件，因此 $\epsilon_{(\vartheta)}(\boldsymbol{x}_{(k)}(\tau), \boldsymbol{y}(\tau), k) := \epsilon_{(\vartheta)}(\boldsymbol{x}_{(k)}(\tau), R(\tau), k)$。这些回报被归一化，以保持 $R(\tau) \in [0, 1]$。对高回报轨迹取样相当于对

算法 1 利用决策扩散器进行条件规划

输入噪声模型 ϵ_θ，逆动力学 f_ϕ，引导尺度 ω，历史长度 C，条件 y 初始化 h 队列（长度= C），$t \leftarrow 0$ // 保持长度为 C 的历史记录
while not done **do**
  观察状态 s；h.insert(s)；初始化 $x_K(\tau) \sim \mathcal{N}(0, \alpha I)$
  for $k = K \ldots 1$ **do**
    $x_k(\tau)[: \text{length}(h)] \leftarrow h$ // 约束计划与历史保持一致
    $\hat{\epsilon} \leftarrow \epsilon_\theta(x_k(\tau), k) + \omega(\epsilon_\theta(x_k(\tau), y, k) - \epsilon_\theta(x_k(\tau), k))$ // 无分类指导
    $(\mu_{k-1}, \Sigma_{k-1}) \leftarrow \text{Denoise}(x_k(\tau), \hat{\epsilon})$
    $x_{k-1} \sim \mathcal{N}(\mu_{k-1}, \alpha\Sigma_{k-1})$
  结束
  从 $x_0(\tau)$ 提取 (s_t, s_{t+1})
  执行 $a_t = f_\phi(s_t, s_{t+1})$；$t \leftarrow t + 1$
结束 while

$R(\tau) = 1$。请注意，我们没有使用任何 Q 值，因为这需要动态编程。

满足约束条件轨迹可能满足各种约束条件，每种约束条件都由集合 c_i 表示，例如达到特定目标、以特定顺序访问状态或避开状态空间的某些部分。为了生成满足给定约束条件 c_i 的轨迹，我们将噪声模型的条件设定为单次编码，即 $\epsilon_\theta(x_k(\tau), y(\tau), k) := \epsilon_\theta(x_k(\tau), \mathbb{1}(\tau \in c_i), k)$。虽然我们使用离线数据集进行训练，其中轨迹只满足一个可用约束条件，但在推理时，我们可以同时满足多个约束条件。

技能组合 技能 i 可由一组演示 β_i 指定。为了生成展示给定技能的轨迹，我们将噪声模型条件化为单次编码，这样 $\epsilon_\theta(x_k(\tau), y(\tau), k) := \epsilon_\theta(x_k(\tau), \mathbb{1}(\tau \in \beta_i), k)$。虽然我们训练的是单个技能，但在推理过程中，我们可以进一步将这些技能组合在一起。

假设我们已经掌握了 n 个不同条件变量的数据分布 $q(x_0(\tau) \mid y^{(1)}(\tau)), \ldots, q(x_0(\tau) \mid y^{(n)}(\tau))$ 对于 n 个不同的条件变量，我们可以从组成的数据分布 $q(x_0(\tau) \mid y^{(1)}(\tau), \ldots, y^{(n)}(\tau))$ 中进行采样（Liu 等人，2022 年）：

$$\hat{\epsilon} := \epsilon_\theta(x_k(\tau), \varnothing, k) + \omega \left(\sum_{i=1}^{n} \epsilon_\theta(x_k(\tau), y^{(i)}(\tau), k) - \epsilon_\theta(x_k(\tau), \varnothing, k) \right) \tag{9}$$

这一特性假定 $\{y^{(i)}(\tau)\}_{i=1}^{n}$ 在状态轨迹 $x_0(\tau)$ 的条件下是独立的。
然而，我们根据经验观察到，只要条件变量的组成是可行的，就不必严格满足这一假设。有关更详细的讨论，请参阅附录 D。我们利用这一特性在测试时将多个约束条件或技能组合在一起。我们还将在附录 J 中展示决策扩散器如何避免特定的约束或技能（NOT）。

3.4 训练决策扩散器

决策扩散器是我们的决策条件生成模型，采用监督方式进行训练。给定一个轨迹数据集 $\mathcal{D}$，每个轨迹都标有其实现的回报、满足的约束或展示的技能，我们同时训练反向扩散过程 p_θ，通过噪声模型 ϵ_θ 和反向动力学模型 f_ϕ 进行参数化，损失如下：

$$L(\theta, \phi) = \mathbb{E}_{k, \tau \in \mathcal{D}, \beta \sim \text{Bern}(p)} [\| \epsilon - \epsilon_\theta(x_k(\tau), (1-\beta) y_\tau + \beta\varnothing, k) \|^2] + \mathbb{E}_{(s, a, s') \in \mathcal{D}} [\| a - f_\phi(s, s') \|^2].$$

对于每条轨迹 τ，我们首先采样噪声 $\epsilon \sim \mathcal{N}(0, I)$ 和一个时间步 $k \sim \mathcal{U}\{1, \ldots, K\}$。然后，我们构建一个有噪声的状态 $x_k(\tau)$ 阵列，最后预测噪声为 $\hat{\epsilon}_\theta := \epsilon_\theta(x_k(\tau), y(\tau), k)$。需要注意的是，由于概率为 p，我们忽略了条件信息，因此逆动态训练使用的是单个过渡而不是轨迹。

架构 我们使用时态 U-Net 架构对 ϵ_θ 进行参数化，该架构是一个由重复卷积残差块组成的神经网络（Janner 等人，2022 年）。这实际上是将一系列状态 $x_k(\tau)$ 视为图像，其中高度代表单个状态的维度，宽度代表单个状态的维度。

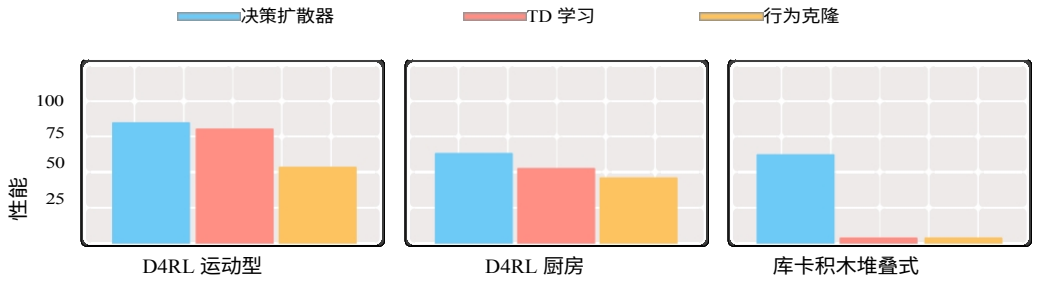


图 4：**结果概览。**在 D4RL 运动任务、D4RL 厨房任务和 Kuka Block Stacking 任务（单一约束条件）中，仅使用条件生成建模目标，Decision Diffuser 的表现优于 TD 学习（CQL）和 Behavorial Cloning（BC）。在性能指标方面，我们对 D4RL 任务（运动和厨房）采用归一化平均回报率（Fu 等人，2020 年），对 Block Stacking 采用成功率。

表示轨迹的长度。我们将条件信息 $y(\tau)$ 编码为标量或一热向量，并通过多层感知器 (MLP) 将其投射到潜在变量 $z \in \mathbb{R}^{(h)}$。当 $y(\tau) = \emptyset$ 时，我们将 z 的条目清零。我们还使用 MLP 对反动态 f_ϕ进行参数化。有关实现细节，请参阅附录 B。

低温采样 在算法 1 的去噪步骤中，我们计算 μ_{k-1}和 Σ_{k-1}从有噪声的状态序列和预测噪声中提取。我们发现，采样 $x_{k-1} \sim \mathcal{N}(\mu_{k-1}, \alpha\Sigma_{k-1})$，其中方差按 $\alpha \in [0, 1]$ 缩放，可得到质量更好的序列（相当于采样较低温度样本）。有关适当的烧蚀研究，请参阅 附录 C。

4 实验

在实验部分，我们探讨了决策扩散器在各种不同决策任务中的功效（如图 4 所示）。特别是，我们评估了**（1）**从离线数据中恢复有效 RL 策略的能力；**（2）**生成满足多组约束条件的行为的能力；**（3）**将多种不同技能组合在一起的能力。此外，我们还从经验上证明了无分类器引导、低温采样（附录 C）和逆动力学（附录 F）的使用，并测试了决策扩散器对随机动力学的鲁棒性 （附录 G）。

4.1 离线强化学习

设置 我们首先测试决策扩散器能否生成收益最大化轨迹。为了测试这一点，我们在公开的 D4RL 数据集（Fu 等人，2020 年）上训练了一个状态扩散过程和反动力学模型。我们与现有的离线 RL 方法进行了比较，包括无模型算法，如 CQL（Kumar 等人，2020 年）和 IQL（Kostrikov 等人，2022 年），以及基于模型的算法，如轨迹转换器（TT，Janner 等人，2021 年）和 MoReL（Kidambi 等人，2020 年）。我们还与决策转换器（DT）（Chen 等人，2021 年）等序列模型和 Diffuser（Janner 等人，2022 年）等扩散模型进行了比较。

结果 在一系列不同的离线强化学习任务中，我们发现决策扩散器要么具有竞争力，要么优于我们的许多离线 RL 基线（表 1）。它还优于 Diffuser 和序列建模方法，如决策变换器和轨迹变换器。在需要长期信用分配的较难的 D4RL Kitchen 任务中，决策扩散器与其他方法的差距变得更加明显。

为了说明无分类器引导的重要性，我们还将 CondDiffuser 与基准 CondDiffuser 进行了比较，后者与 Diffuser 一样，在无分类器引导的情况下对状态和动作序列进行扩散。在表 2 中，我们发现 CondDiffuser 在 3 个环境中的 2 个比 Diffuser 更优。决策扩散器比 CondDiffuser 更进一步，在所有 3 种环境中都表现得更好。我们的结论是，学习逆动力学是对行动进行扩散的一个很好的替代方案。我们将在附录 F 中进一步分析何时使用逆动力学，何时对行动进行扩散。我们还与 CondMLPDiffuser 进行了比较。

根据以状态和返回值为条件的扩散过程对动作进行去噪。我们发现，在各种扩散模型中，CondMLPDiffuser 的表现最差。到目前为止，我们主要测试了具有确定性（或接近确定性）环境动态的离线 RL 任务。因此，我们在附录 G 中测试了决策扩散器对随机动态的鲁棒性，并在改变环境动态的随机性时与扩散器和 CQL 进行了比较。最后，我们在附录 E 中分析了决策扩散器的运行时 特性。

4.2 约束满足

设置 接下来，我们将使用图 5 所示的库卡积木堆叠环境（Janner 等人，2022 年）来评估我们生成满足一系列约束条件的轨迹的能力。在这个领域中，有四个积木块，它们可以*堆叠*成一个塔，也可以*重新排列*成多个塔。BlockHeight(i)> BlockHeight(j) 这样的约束条件要求将积木 i 放置在积木 j.我们从 10,000 个专家演示中训练决策扩散器，每个演示都满足以下条件之一约束条件。我们将这些区块的位置随机化，并在推理时考虑两项任务：采样满足数据集中之前出现过的单个约束条件的轨迹，或满足一组从未提供过演示的约束条件的轨迹。在后一种情况下，我们要求决策扩散器生成轨迹，使 BlockHeight(i)> BlockHeight(j)> BlockHeight(k) 满足四个区块 i、j、k 中三个的要求。更多详情，请参阅附录 H。

结果 在堆叠和重新排列设置中，决策扩散器满足单一约束的成功率都高于扩散器（表 3）。我们还与 BCQ（藤本等人，2019 年）和 CQL（库马尔等人，2020 年）进行了比较，但它们始终无法堆叠或重新排列区块，导致成功率为 0.0。与这些基线不同，我们的方法可以根据等式 9 有效地同时满足多个约束条件。有关这些生成轨迹的可视化，，请访问网站 https://anuragajay.github.io/decision-diffuser/。

4.3 技能构成

设置 最后，我们来看看如何将不同的技能组合在一起。我们考虑了 Unitree-go- running 环境（Margolis 和 Agrawal，2022 年），在该环境中，可以发现一个四足机器人以各种步态奔跑，如绑定、踱步和小跑。我们探索是否有可能只对单个步态进行训练，就能生成在这些步态之间转换的轨迹。我们为每种步态收集了 2500 个演示数据集，并在此基础上训练决策扩散器。

结果 在测试过程中，我们根据方程 9 使用反向扩散过程的噪声模型，对具有全新运行行为的四足机器人轨迹进行采样。图 6 显示了一条以束缚开始但以踱步结束的轨迹。附录 I 提供了更多可视化的奔跑步态组合。虽然从视觉上看，轨迹

数据集	环境	BC	CQL	IQL	DT	TT	其他	扩散器	DD
医疗专家	半边天	55.2	91.6	86.*7*	86.8	**95**	53.3	79.8	90.6± *1.3*
医疗专家	霍普	*52.5*	105.4	91.5	107.6	**110.0**	*108.7*	107.2	**111.8**± *1.8*
医学专家	沃克2d	**107.5**	**108.8**	**109.6**	**108.1**	101.9	95.6	**108.4**	**108.8**± *1.7*
中型	半边天					46.9	42.1	44.2	**49.1**± *1.0*
中型	料斗		58.5	*66.3*			**95.4**	58.5	79.3± *3.6*
中型	沃克2d	75.*3*	72.5	78.3		79	*77.8*	79.7	**82.5**± *1.4*
Med-Replay	HalfCheetah	36.6	**45.5**	**44.2**	36.6	*41.9*	40.2	42.2	39.3± *4.1*
Med-Replay	料斗	18.1	95	94.7	82.7	91.5	*93.6*	96.8	**100**± *0.7*
Med-Replay	Walker2d	26.0	77.*2*	*73.9*	66.6	**82.6**	49.8	61.2	75± *4.3*
平均值		51.9	77.6	77	74.7	78.9	72.9	75.3	**81.8**
混合型	厨房		52.4	51	-	-	-	-	**65**± *2.8*
部分	厨房	38	50.1	46.3	-	-	-	-	**57**± *2.5*
平均值		44.8	51.2	48.7	-	-	-	-	**61**

表 1：**离线强化学习性能。**我们发现，在 D4RL 任务中，决策扩散器（DD）的归一化平均收益与当前离线强化学习方法不相上下，甚至更胜一筹（Fu 等人，2020 年）。我们报告了 5 个随机种子的平均值和标准误差。

Hopper-*	扩散器 CondDiffuser	CondMLPDiffuser	决策扩散器
中间专家	107.6	**111.3**	105.6	**111.8**± 1.6
中型	58.5	*66.3*		**79.3**± 3.6
中型	*96.8*	76.5	66.5	**100**± 0.7

表 2：**消融。**在 Diffuser 中使用无分类器引导，从而产生 CondDiffuser，提高了 2 个（共 3 个）环境中的性能。此外，在 Decision Diffuser 中使用反动力学进行行动预测，可提高在所有 3 种环境中的性能。而根据当前状态和目标返回值对当前行动进行扩散的 CondMLPDiffuser 的表现则不尽如人意。

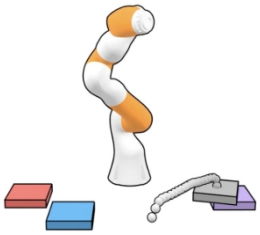
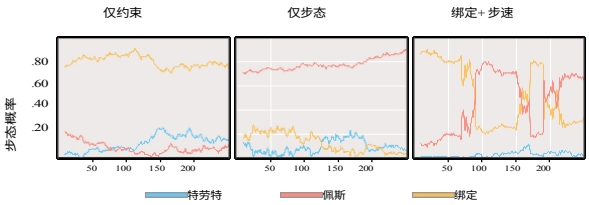


图 5：**Kuka 块堆叠任务。**

环境	扩散器	DD
单一约束 - 堆叠	45.6± 3.1	**58.0**± 3.1
单一约束 - 重新排列	58.9± 3.4	**62.7**± 3.1
单一约束平均值	52.3	**60.4**
多重约束 - 叠加	-	**60.3**± 3.1
多重约束 - 重新排列	-	**67.2**± 3.1
多重制约平均	-	**63.8**

表 3：**通过约束最小化实现块堆叠。**与 Diffuser 相比，Decision Diffuser（DD）在生成满足一组块堆叠约束条件的轨迹的成功率方面有所提高。它还能在测试期间灵活组合多个约束条件。我们报告了 5 个随机种子的平均成功率和标准误差。

使用决策扩散器生成的轨迹包含一种以上的步态，我们希望准确量化不同步态的组合效果。为此，我们对分类器进行了训练，以预测轨迹中每个时间步或帧的四足动物的奔跑步态（即奔跑、踱步或小跑）。我们重复使用为训练决策扩散器而收集的演示来训练该分类器，其中我们的输入被定义为固定时间段内的机器人关节状态（即长度为 10 的状态子序列），而标签则是该序列中演示的步态。步态分类程序的完整细节见附录 I。



条件	速度	步伐	绑定
仅绑定	0.8	1.0	98.2
仅佩斯	1.4	97.7	0.9
绑定+ 步幅	1.4	38.5	60.1

图 7：**跑步步态分类。**分类器会在每个时间步预测四足动物的奔跑步态。在以单一技能为条件生成的轨迹上，如仅以束缚或踱步为条件，分类器以最大概率预测出相应的步态。当以两种技能为条件时，一些时间步被归类为绑定，而另一些时间步则被归类为踱步。

我们以两种方式使用我们的跑步步态分类器：评估四足动物的行为在单个生成轨迹过程中的变化情况，以及测量每种步态在多个生成轨迹中出现的频率。在前者中，我们首先从决策扩散器中抽取三个轨迹样本，条件是束缚步态、踱步步态或两者兼而有之。对于每条轨迹，我们分别绘制出每种步态在序列长度上的分类概率。如图 7 所示，分类器分别预测束缚步态和踱步步态是在此条件下采样的轨迹中最有可能出现的跑步步态。当轨迹是以两种步态为条件生成时，分类器会从预测概率最大的一种步态过渡到另一种步态。事实上，在一些情况下，四足动物的行为会根据分类器在束缚和踱步之间切换。这与图 6 中报告的可视化效果一致。在图 7 所示的表格中，我们考虑了使用决策扩散器生成的 1000 条轨迹，这些轨迹是以所列的一种或两种步态为条件的。我们记录了四足动物的奔跑步态被分类为小跑、踱步或束缚的时间比例。事实证明，分类器

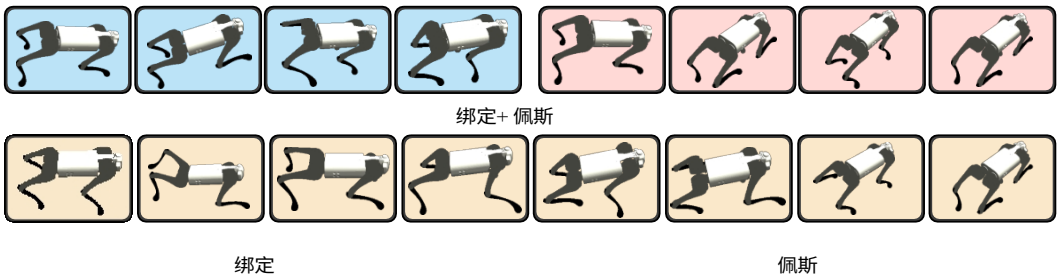


图 6：**组合运动技能。**决策扩散器可利用专家示范模仿个人跑步步态，并在测试期间将多种不同技能组合在一起。可在 https://anuragajay.github.io/decision-diffuser/ 上观看的视频最能说明结果。

当通过组合两种步态对轨迹进行采样时，有 38.5% 的时间将行为识别为束缚，另外 60.1% 的时间将行为识别为踱步。这证实了这样一个事实，即尽管决策扩散器 只接受过单个步态的训练，但它确实可以组成跑步行为。

5 相关工作

扩散模型 扩散模型在学习图像和文本数据的生成模型方面大有可为（Saharia 等人，2022 年；Nichol 等人，2021 年；Nichol & Dhariwal，2021 年）。它将数据采样过程表述为一个迭代去噪过程（Sohl-Dickstein 等人，2015 年；Ho 等人，2020 年）。去噪过程也可解释为对数据分布的梯度进行参数化（Song 等人，2021 年），优化分数匹配目标（Hyva¨rinen，2005 年），因此也可解释为基于能量的模型（Du & Mordatch，2019 年；Nijkamp 等人，2019 年；Grathwohl 等人，2020 年）。为了生成以某些附加信息（如：文本）为条件的数据样本（如：图像），先前的研究（Nichol & Dhariwal, 2021）已经学习了一个分类器来促进条件采样。最近的研究（Ho & Salimans, 2022）认为，利用隐式分类器的梯度（由有条件模型和无条件模型的得分函数之差形成）来促进条件采样。结果表明，与基于分类器的引导相比，无分类器引导能生成更好的条件样本。上述所有工作主要集中在文本或图像的生成上。最近的研究还利用扩散模型来模仿人类行为（Pearce 等人，2023 年），以及离线 RL 中的参数化策略（Wang 等人，2022 年）。Janner 等人（2022 年）利用无条件扩散模型生成了由状态和行动组成的轨迹，因此需要在有噪声的状态-行动对上训练奖励函数。在推理过程中，估计的奖励函数会引导反向扩散过程向高回报轨迹样本发展。相比之下，我们并不单独训练奖励函数或扩散过程，而是用一个单一的条件生成模型对数据集中的轨迹进行建模。这就确保了所学扩散过程的采样程序在推理时与训练时相同。

奖励条件政策 之前的研究（Kumar 等人，2019；Schmidhuber，2019；Srivastava 等人，2019；Emmons 等人，2021；Chen 等人，2021）已经研究了通过奖励条件行为克隆学习奖励条件政策。Chen 等人（2021 年）使用变压器（Vaswani 等人，2017 年）对奖励条件策略进行建模，并获得了与离线 RL 方法相当的性能。Emmons 等人（2021 年）在没有使用转换器策略的情况下，获得了与 Chen 等人（2021 年）类似的性能，但依赖于对 MLP 策略进行仔细的容量调整。与这些研究不同的是，除了对回报建模外，Decision Diffuser 还能对约束或技能建模，并在测试期间通过灵活组合多个约束或技能生成新的行为。

6 讨论

我们提出的决策扩散器（Decision Diffuser）是一种用于顺序决策的条件生成模型。它将离线顺序决策制定构建为条件生成模型，避免了强化学习的需要，从而使决策制定管道更加简单。通过对高回报进行采样，

它能够捕捉数据集中的最佳行为，并在标准基准（如 D4RL）上优于现有的离线 RL 方法。除了回报，它还可以

在测试过程中灵活组合约束条件或合成技能，从而生成新的行为。

在这项工作中，我们专注于离线顺序决策，从而避免了探索的需要。利用 Zheng 等人（2022 年）的观点，未来的工作可以通过利用状态序列模型的熵进行探索，研究决策扩散器的在线微调。虽然我们的工作侧重于基于状态的环境，但也可以像 Rombach 等人（2022 年）所做的那样，通过在潜空间而非观测空间进行扩散，将其扩展到基于图像的环境。有关决策扩散器局限性的详细讨论，请参阅附录 M。

致谢

作者感谢 Ofir Nachum、Anthony Simeonov 和 Richard Li 对本论文早期草稿的有益反馈；感谢 Jay Whang 和 Ge Yang 就无分类器指导进行的讨论；感谢 Gabe Margolis 在单元树实验中提供的帮助；感谢 Micheal Janner 提供 Kuka 块堆叠的可视化代码；感谢 Improbable AI Lab 成员的讨论和有益反馈。我们感谢麻省理工学院超级云和林肯实验室超级计算中心提供的计算资源。本研究得到了国家自然科学基金研究生奖学金、DARPA 机器常识基金、ARO MURI 基金 W911NF-21-1-0328 和 MIT-IBM 基金的支持。

本研究还得到了美国空军研究实验室和美国空军人工智能加速器的部分赞助，并根据合作协议编号 FA8750-19- 2-1000 完成。本文件中的观点和结论仅代表作者本人，不应被解释为代表美国空军或美国政府明示或暗示的官方政策。美国政府有权为政府目的复制和分发重印本，尽管此处有任何版权说明。

作者贡献

Anurag Ajay 构想了将决策视为条件扩散生成模型的框架，实现了决策扩散器算法，进行了离线 RL 和技能合成实验，并协助撰写论文。

杜一伦帮助构思了将决策视为条件扩散生成模型的框架，进行了约束满足的实验，帮助撰写论文，并为 Anurag 提供了建议。

阿比-古普塔（Abhi Gupta）帮助进行了离线 RL 和技能合成实验，参与了研究讨论，并在论文撰写和图表制作中发挥了主导作用。

Joshua Tenenbaum 参与了研究讨论。

Tommi Jaakkola 参与了研究讨论，并建议进行跑步步态分类实验。

Pulkit Agrawal 参与了研究讨论，提出了与动态编程相关的实验建议，并就论文写作、作品定位和整体建议提供了反馈意见。

参考文献

Pulkit Agrawal、Ashvin V Nair、Pieter Abbeel、Jitendra Malik 和 Sergey Levine。通过 "戳 "学会 "戳"：直观物理学的经验学习。*神经信息处理系统进展》*，2016年第29期。

Anurag Ajay、Aviral Kumar、Pulkit Agrawal、Sergey Levine 和 Ofir Nachum。Opal：*ArXiv preprint arXiv:2010.13611*, 2020.

Tom Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared D Kaplan、Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、Amanda Askell、Sandhini Agarwal、Ariel