行为克隆及衍生方法（BC/IBC/PC）

模仿学习，就是要训练 agent 从专家数据中学习人类专家策略，进而达到模仿专家行为的目的。Behavior Cloning（BC）是模仿学习中重要的一类算法，本文将简单介绍 BC 类基础算法以及其变体 Implicit Behavior Cloning（IBC）和 Procedure Cloning（PC）算法。

Behavior Cloning (BC)

行为克隆 (Behavior Cloning) 是一种纯监督学习的方法。我们把专家数据，拆分成“状态-动作”对 (s, a) 以后，看起来就是像监督学习中有标记的数据。让机器学习专家决策行为的直观想法是，用一种监督学习的方式，可以把这个状态作为我们监督学习里面的样本，动作作为监督学习里面的标记，把我们的状态当成神经网络的输入，把神经网络输出当成动作，最后获得学习状态和动作之间的相对应关系。假设我们先采集得到专家数据集 D_*，并从中采样得到状态-动作对 $(s, a) \sim \mathcal{D}_*$，最小化极大似然估计目标进行优化：

$$J_{\mathrm{BC}}(\pi) := \mathbb{E}_{(s,a)\sim\mathcal{D}_*}\left[-\log \pi(a \mid s)\right]$$

BC 的优点是原理易懂，操作简单高效。然而，在监督学习中，我们总是假设数据分布是不会变的，但是在 BC 中，由于要解决的问题是序列决策问题，如果采集的专家数据样本不足，每一步产生的误差会不断的发生累积，最终会越来越偏离原始数据轨迹，产生更大的误差，也被称为 compounding error。

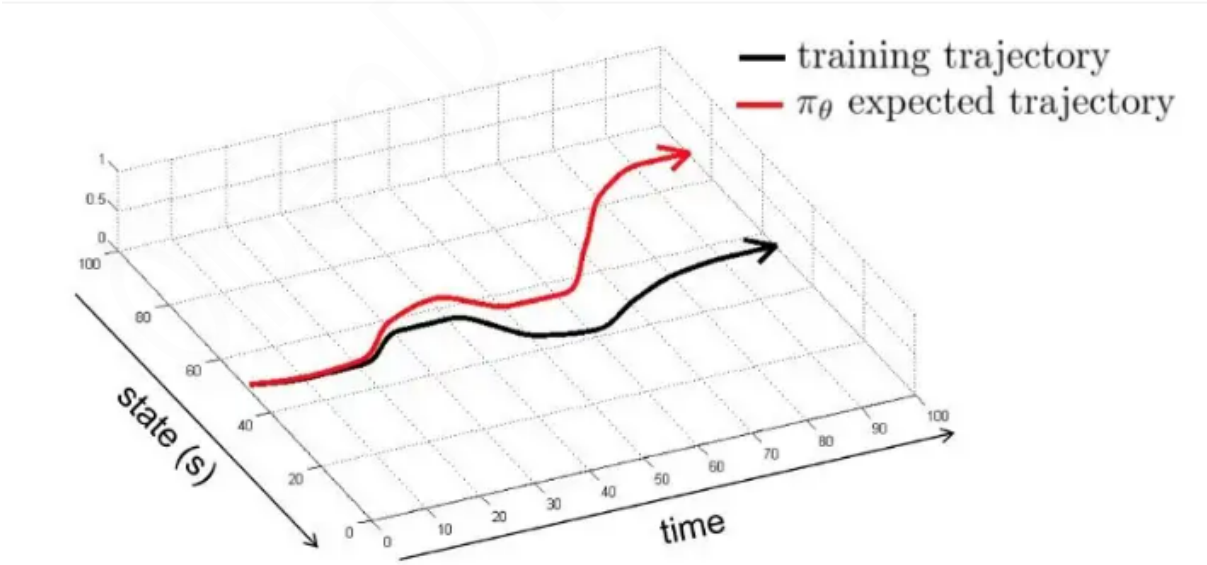

图1 BC的复合误差 (compounding error)

Implicit Behavior Cloning (IBC)

虽然 BC 的原理简单，在实际应用中也取得了不错的表现，但在开发新的模仿学习算法的过程中，BC 模型的基本假设被忽略了，默认将策略本身的形式假设为和其他监督学习方法一样，输入 observation 到输出 action 的直接映射的显式前向模型：

$$\hat{\mathbf{a}} = F_\theta(o)$$

IBC (Implicit Behavior Cloning) 提出用隐式模型 (implicit model) 来重新定义 BC 的基础模型，其策略形式表示为:

$$\hat{\mathbf{a}} = \underset{\mathbf{a}\in\mathcal{A}}{\operatorname{argmin}} E_\theta(\mathbf{o}, \mathbf{a}) \quad \text{instead of} \quad \hat{\mathbf{a}} = F_\theta(\mathbf{o}).$$

它将模仿学习问题描述为一个基于条件能量模型 (energy-based modeling, EBM) 的问题（如图2 (b)），在给定观察 o 时，通过隐式回归得到最优动作 a 。通过理论和实验证明，这一改变可以使 IBC 在非连续性函数、多值函数的拟合效果更好，从而提升模型在更多任务中的性能。

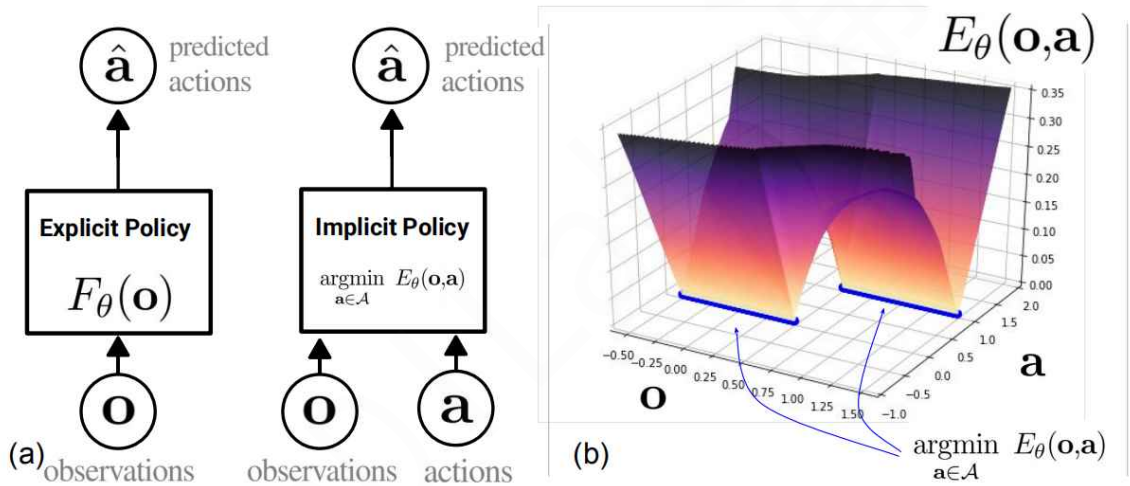

图2 (a) IBC的隐式模型，(b) 能量密度函数图

除此之外，隐式模型也被用在各种强化学习的部分组件，如离散动作空间的动作值函数其实本质上也是一种隐式模型。

IBC 和 BC 对比

我们考察 IBC 和 BC 在简单视觉坐标回归任务上的效果差异。如图3 (a) 所示，实验给定一个带有绿点的彩色图像作为输入，预测绿点的坐标。在这一任务上，模型非常容易过拟合，如何使得模型在训练集的凸包之外有一定泛化能力，是简单显示监督训练的卷积网络众所周知的难题。为了对比显式模型和隐式模型的效果，我们采取如 (b) 所示的相同模型结构，探究在训练数据的凸包以外，模型能否准确预测点的坐标，具有一定的外推能力。

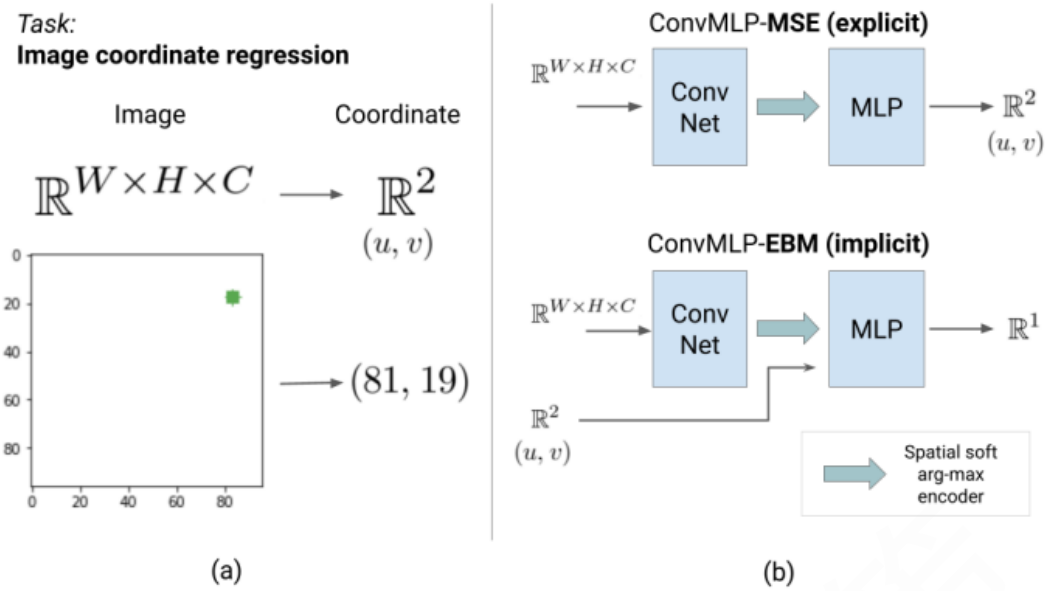

图3 (a)坐标回归任务，(b)BC和IBC模型

如图4所示，图中灰色交叉点为训练数据，灰色虚线显示了训练数据的凸包，在训练数据只有10个时，MSE 显式模型 (explicit model) 在凸包内插值和凸包外外推性能较差（左上），在 30 个训练样本的情况下，MSE 才能够在凸包内合理插值，但在凸包外的预测准确率仍然较低（右上）。相反，采用 EBM 隐式模型 (implicit model) ,convMLP-EBM 模型可以在模型数量较少（10个）时，仍然可以在凸包内和凸包外获得较好的插值和外推效果。

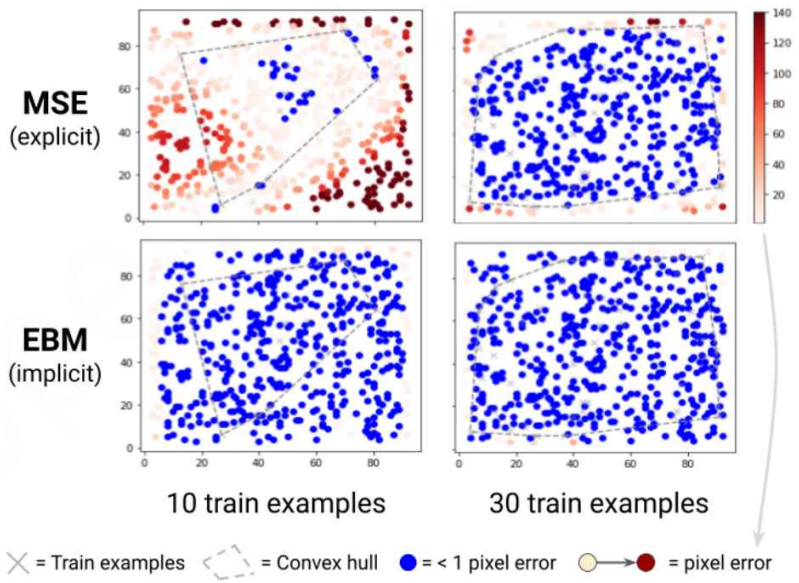

图4 BC和IBC预测准确率对比

将训练样本数量与测试数据预测的 MSE 用折线图表示，我们可以看到，ConvMLP-EBM 隐式模型比 ConvMLP-MSE 显式模型在取得相同的预测效果时，需要的训练数据降低了1到2个数量级，极大的节约了专家数据采集的成本，降低运算量。

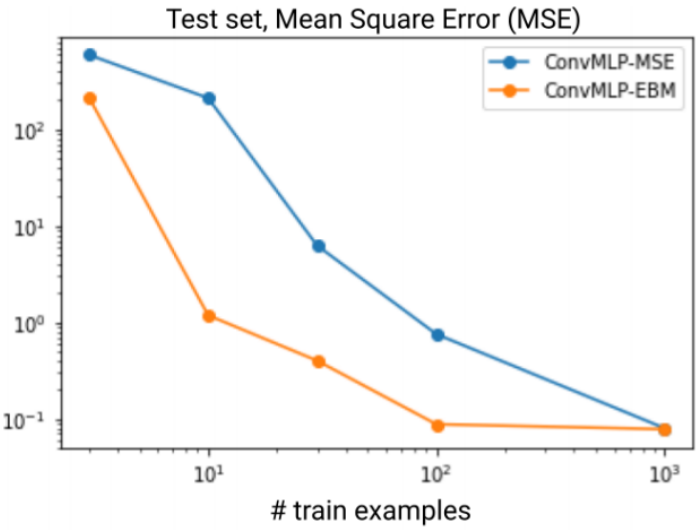

图5 BC和IBC预测误差-训练数据量 性能对比

Procedure Cloning (PC)

传统 BC 将模仿学习视作监督学习，用函数拟合从输入 observation 到输出 action 的映射，从而从专家数据中直接提取策略。但是，在某些问题中，如果专家数据不仅包括 observation 到 action 的映射，还为专家行为提供了更丰富的信息，如路径导航、机器人控制或者策略游戏等问题，通过规划、搜索等多步算法，不仅包含了最终要模仿的输出动作，还包含了如何确定该动作的过程。如果仅采用 BC 这样端到端的学习，就会将问题过分简单化。所以Procedure Cloning (PC) 提出，比起让智能体记住什么状态下采取什么动作，不如让智能体同样学习动作的推理过程，这就是PC的算法动机。

所以为了让智能体学会推理，我们就必须在训练样本的层面进行改进。之前 BC 需要学习的样本是一个 transition，只包含了 (s, a) 两个元素。但是PC的一个样本则包含 (s, x, a) 三个元素。其中 x 是一个向量，它包含了专家 policy 做决策时的中间思考结果。正是因为 PC 在训练样本中引入了专家的中间推理信息，因此它不仅能够模仿专家的动作，也能模仿专家的推理过程，进而得到更好的泛化性表现。

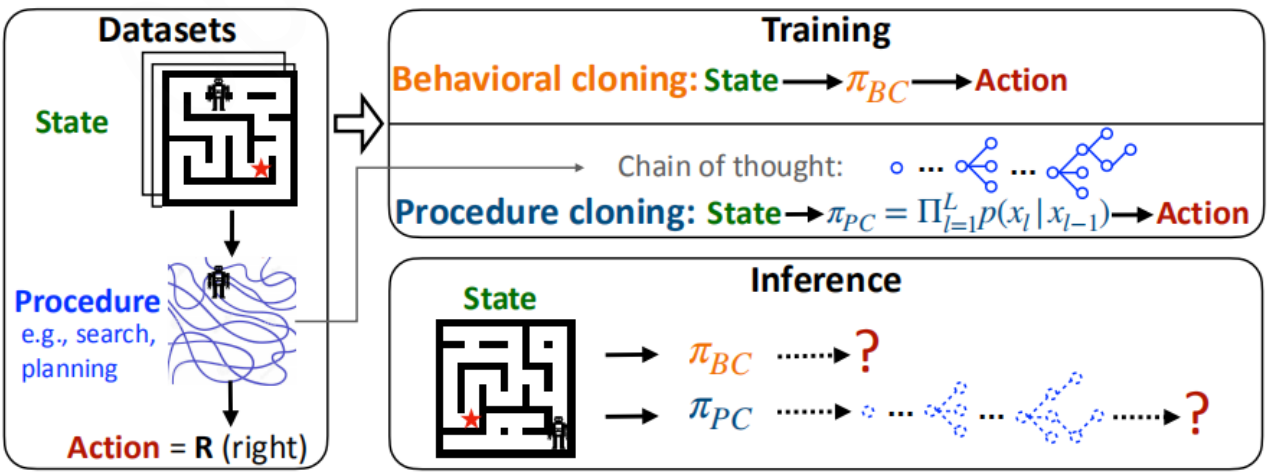

图6 PC算法流程图

PC 和 BC/Auxiliary BC 对比

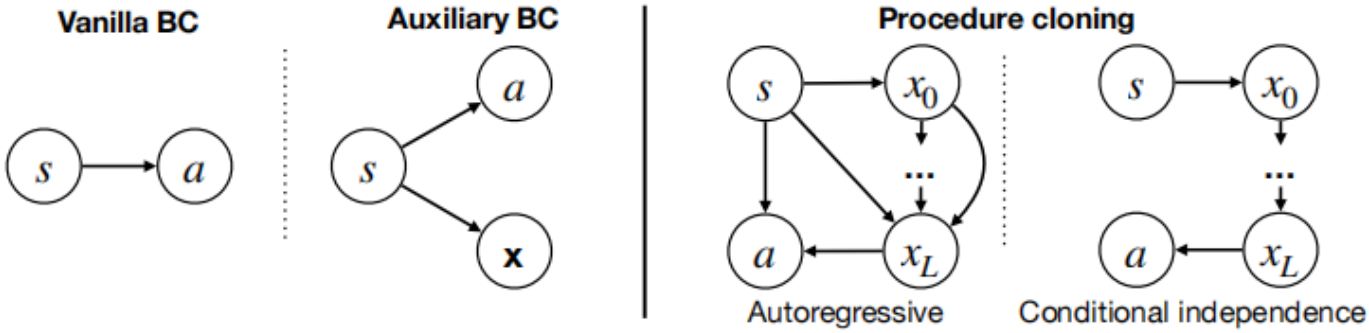

图7 PC 和 BC/Auxiliary BC 对比

我们将原始 BC (vanilla BC)，辅助BC (Auxiliary BC)，以及 PC 三种模仿学习对比：

- Vanilla BC：输入状态 s ，通过监督学习直接拟合得到目标动作 a 。
- Auxiliary BC： 输入仍然是状态 s ，但是此时需要拟合的输出不仅有相应的动作 a ，也有专家的辅助中间结果 $\mathbf{x}$ 。这里的 $\mathbf{x}$ 和 a **相互独立**，但都依赖于 s ，通过增加一个拟合辅助中间状态 $\mathbf{x}$ 任务，增加一个辅助损失，成功地在算法中引入中间推理信息了。但是，这样做更像是对网络训练施加了某种正则化，而不能让网络直接学习如何像专家那样推理，没有体现中间的推理过程。

$$p(\mathbf{a}, \mathbf{x} \mid s) = \pi(a \mid s) \cdot p(\mathbf{x} \mid s)$$

- PC：不同于Auxiliary BC，PC 侧重于 $\mathbf{x}$ 和 a 条件独立假设不成立的情况，将过程信息 $\mathbf{x}$ 看作 a 的前提。并且延伸出两种形式，分别是自回归和条件独立：
  - 自回归的形式：用状态 s 预测专家思考的第一个中间状态 x_0 ， 然后再由 $\{s, x_0\}$ 预测下一个中间思考状态 x_1 ，再由 $\{s, x_0, x_1\}$ 预测 x_2 。以此类推，用 $\{s, x_0, x_1, \ldots, x_L\}$ 预测 x_L ，并最终预测相应的动作 a 。
  - 条件独立的形式：首先网络由状态 x 预测专家思考的第一个中间状态 x_0 ，然后再由 x_0 预测下一个中间思考状态 x_1 。以此类推，直到最后预测动作 a 。

PC 进一步的实验分析

PC 的算法动机是学习动作的推理过程，所以可以结合许多类型的搜索或者多步算法学习，如广度优先搜索 (BFS) 、蒙特卡洛树搜索 (MCTS) 等具有一定推理过程的算法，我们可以通过两个实例实验来详细了解 PC 的学习过程。另外，PC 的各个预测步中都基于 IBC 中提出的隐式模型形式来输出最终结果。

迷宫实验：PC+BFS

广度优先搜索 (BFS) 是一种常用的路径规划算法，我们使用一个网格化的迷宫环境，在这个环境中，智能体可以执行四个离散的动作，包括上下左右移动，使得智能体从起始位置导航到目标位置。

专家 BFS 先从起点执行一遍搜索，获得每个网格到迷宫终点的最优路线，再从迷宫的终点处开始回溯。BFS搜索过程中维护一张 visit map 来记录该网格是否已被搜索访问，具体执行的动作，以及该位置是否有被回溯，我们只需要将 BFS 搜索过程中网格的中间观察值记录下来，得到一系列过程数据 $\mathbf{x} = (x_1, \ldots, x_L)$。同时我们注意到由于每个中间过程 x_l 值与前一个中间观察值 x_{l-1} 有关，所以这里的PC我们采用条件独立形式的 PC：

$$p(a, \mathbf{x} \mid s) = p(a \mid x_L) \cdot \Pi_{l=1}^{L} p(x_\ell \mid x_{\ell-1}) \cdot p(x_0 \mid s)$$

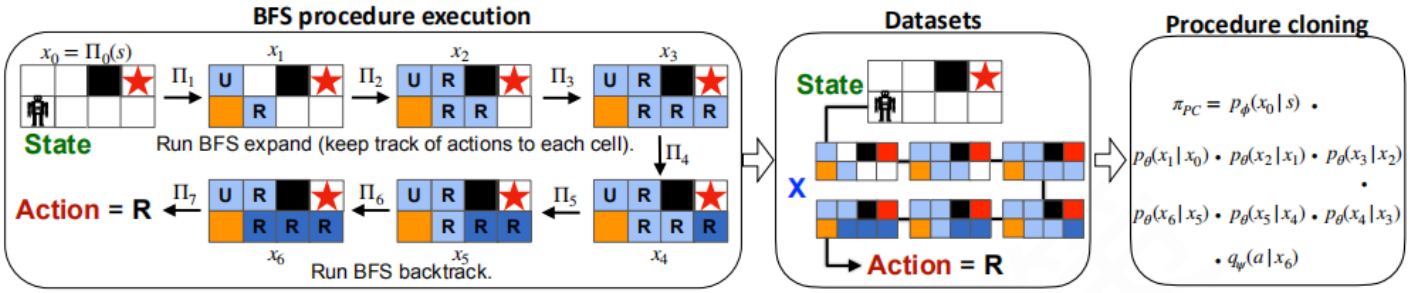

图8 PC+BFS算法流程

最终实验结果如下，在不同大小的迷宫实验中，对比 PC、BC、Auxiliary BC (Aux BC) 以及应用随机裁剪、平移缩放等方式进行数据增强后的 BC (Aug BC) 进行对比，横坐标表示训练的迷宫数，纵坐标表示从随机位置到达目标点的平均成功率，每个迷宫提供1、4和16条专家轨迹训练。可以看到相比于 BC、AuxBC、Aug BC、PC 能够在较少的专家数据的情况下学习到较高的成功率，且随着迷宫增大，PC也可以获得较好的泛化性。Aux BC 和 Aug BC在较小的迷宫中成功率较高，但在较大的成功率较差。

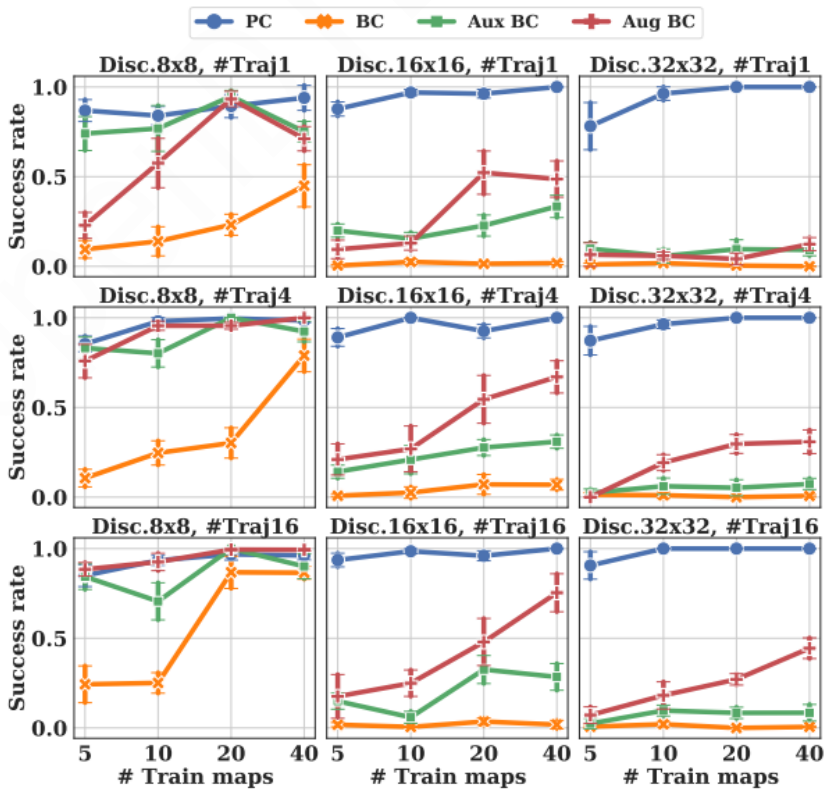

图9 PC、BC、AuxBC、Aug BC迷宫成功到达率对比

MinAtar 实验：PC + MCTS

MinAtar 是 Atari 游戏的缩小版，原论文采用 AlphaZero 风格的蒙特卡洛树搜索算法 (AlphaZero-style Monte-Carlo tree search / MCTS) 收集专家轨迹，由于这里的每个状态都依赖于之前所有状态，所以 PC 采取自回归 (autoregressive) 形式 PC：

$$p(a, \mathbf{x} \mid s) = p\left(a \mid \mathbf{x}, s\right) \cdot \Pi_{l=1}^{L} p\left(x_{\ell} \mid \mathbf{x}_{<\ell}, s\right) \cdot p\left(x_0 \mid s\right)$$

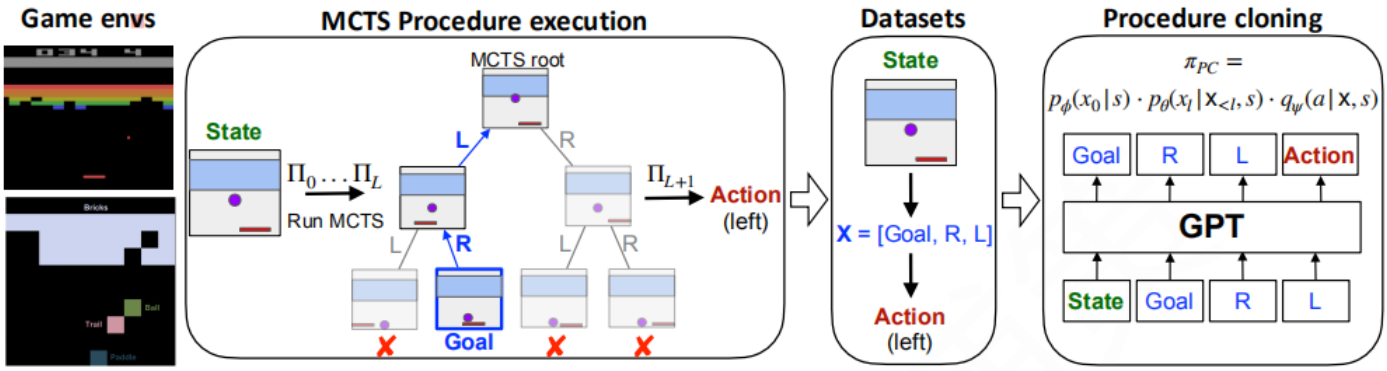

图10 PC + MCTS 算法流程

PC+MCTS 的算法过程如上图所示，由于在 MinAtar 中运行 MCTS 比较复杂，需要进行多次 MCTS 模拟（$\Pi_0, \ldots,$ Π_L），每次模拟都会包括选择、扩展、模拟和回溯过程以及不同的树结构，所以要想捕获完整的搜索的过程数据十分困难。原论文从当前状态节点出发，仅记录了最后一次 MCTS 模拟之后的产生的最佳搜索轨迹 Π_L：$[L, R, Goal]$，由于最终预测目标为当前执行的 Action，所以我们将这个轨迹的逆顺序 $\mathbf{x} = [Goal, R, L]$ 作为过程数据，x_0 就是最终的目标图像 Goal，先预测 x_0，然后按照逆序，使用类似语言模型 GPT 一样的自回归模型，从目标图像开始向前预测，最终预测得到当前的最佳执行动作。

最终，原论文的实验是在 MinAtar 的一般环境上采集专家数据训练，并分别在与训练环境相同配置的环境、0.1概率增加粘滞动作的环境以及增加游戏难度的三种环境对算法进行测试，对比PC、BC、AuxBC、Aug BC的平均奖励，PC在多种环境中均表现出最好的性能。更多实验细节，大家可以参考原论文第6.3节及附录部分。

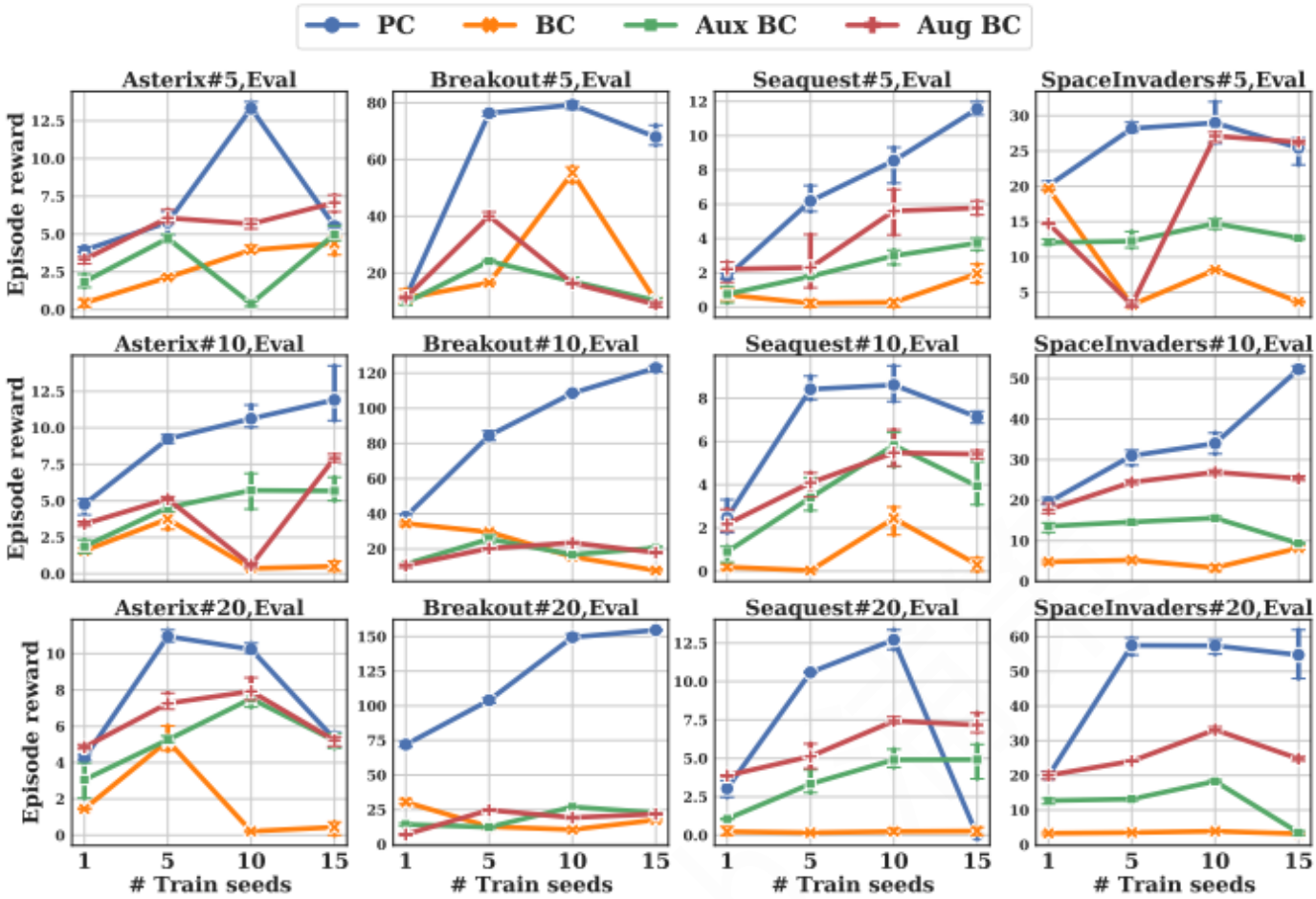

图11 MinAtar不同游戏中，PC、BC、AuxBC、Aug BC的平均奖励对比

参考文献：

BC：Zheng, B., Verma, S., Zhou, J., Tsang, I.W., & Chen, F. (2021). Imitation Learning: Progress, Taxonomies and Challenges. *IEEE transactions on neural networks and learning systems, PP*.

IBC: Yang, M., Schuurmans, D., Abbeel, P., & Nachum, O. (2022). Chain of Thought Imitation with Procedure Cloning. *ArXiv, abs/2205.10816*.

PC: Florence, P.R., Lynch, C., Zeng, A., Ramirez, O., Wahid, A., Downs, L., Wong, A.S., Lee, J., Mordatch, I., & Tompson, J. (2021). Implicit Behavioral Cloning. *ArXiv, abs/2109.00137*.