🧠 世界模型深度解析

从 World Models 到 Dreamer 再到未来

在梦中学习：让 AI 拥有想象力

📋 分享大纲

1. **为什么需要世界模型？** - 核心动机与直觉
2. **World Models (2018)** - 开山之作
3. **Dreamer 系列 (2020-2023)** - 工程化突破
4. **核心技术深度剖析** - RSSM、KL Balancing、离散潜在空间
5. **新一代方向** - Genie、JEPA、统一架构
6. **总结与展望**

第一部分

🎯 为什么需要世界模型?

人类如何学习?

真实体验

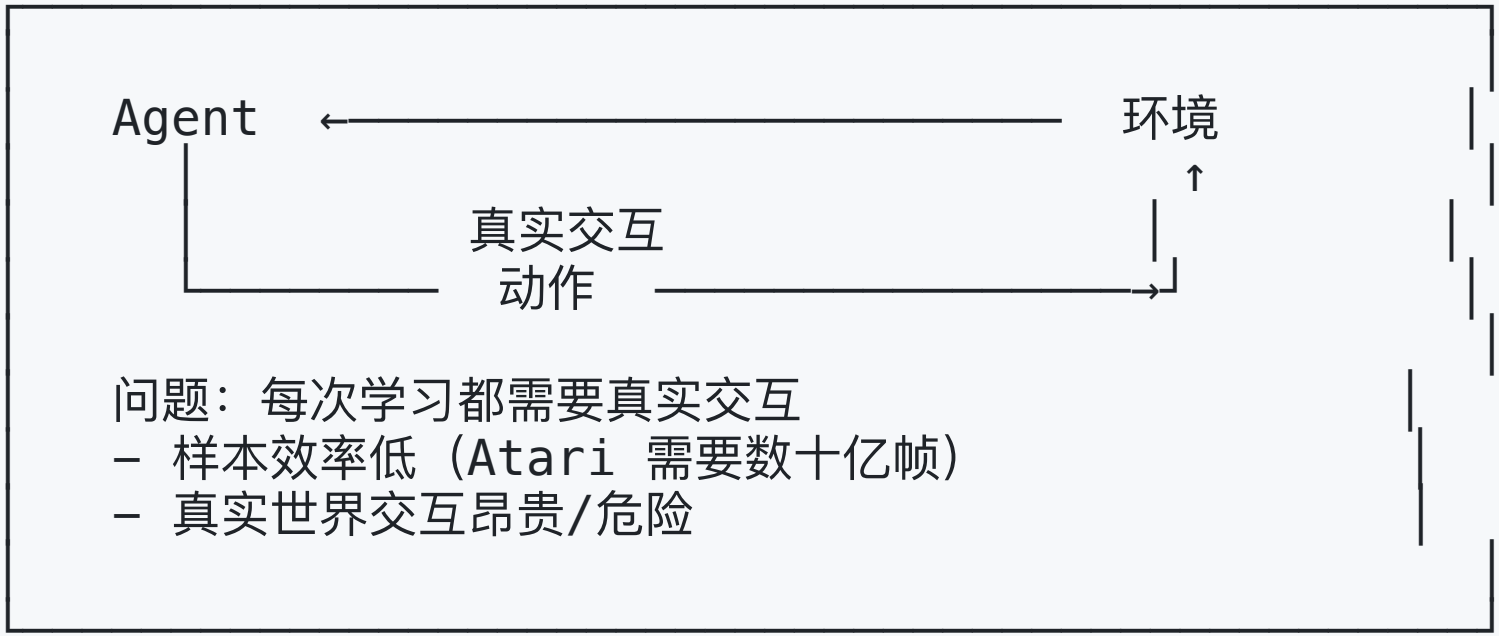
- 学开车：上路练习
- 学游泳：下水尝试
- 成本高、有风险

> 爱因斯坦：想象力比知识更重要

想象练习 ✨

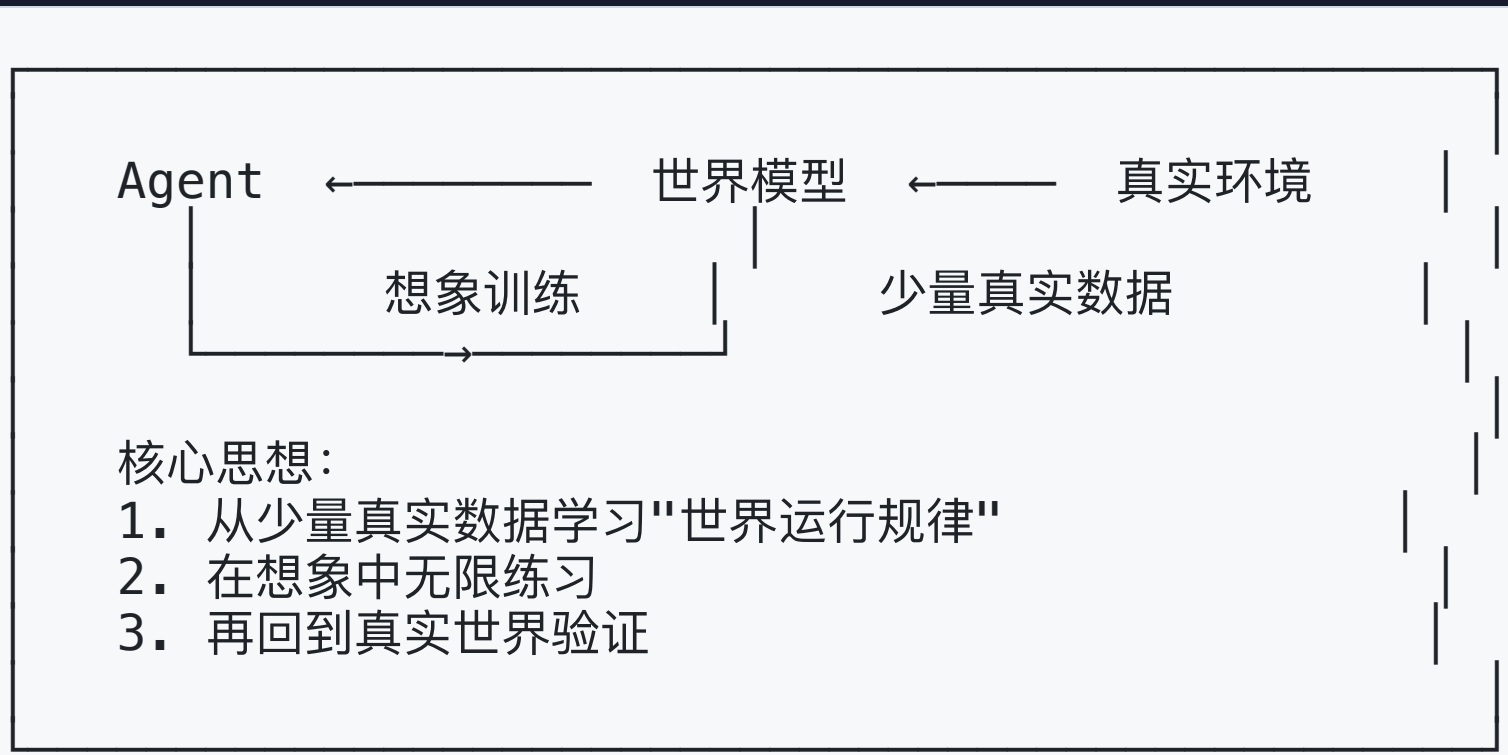
- 脑中预演路线
- 想象动作要领
- **零成本、无风险**

传统强化学习的困境



问题：每次学习都需要真实交互
- 样本效率低（Atari 需要数十亿帧）
- 真实世界交互昂贵/危险

世界模型的解决方案



核心思想：
1. 从少量真实数据学习"世界运行规律"
2. 在想象中无限练习
3. 再回到真实世界验证

形式化定义

世界模型 = 学习环境的转移函数

$$\hat{s}_{t+1}, \hat{r}_t = f_\theta(s_t, a_t)$$

给定当前状态 s_t 和动作 a_t，预测下一状态 $\hat{s}_{t+1}$ 和奖励 $\hat{r}_t$

关键能力：

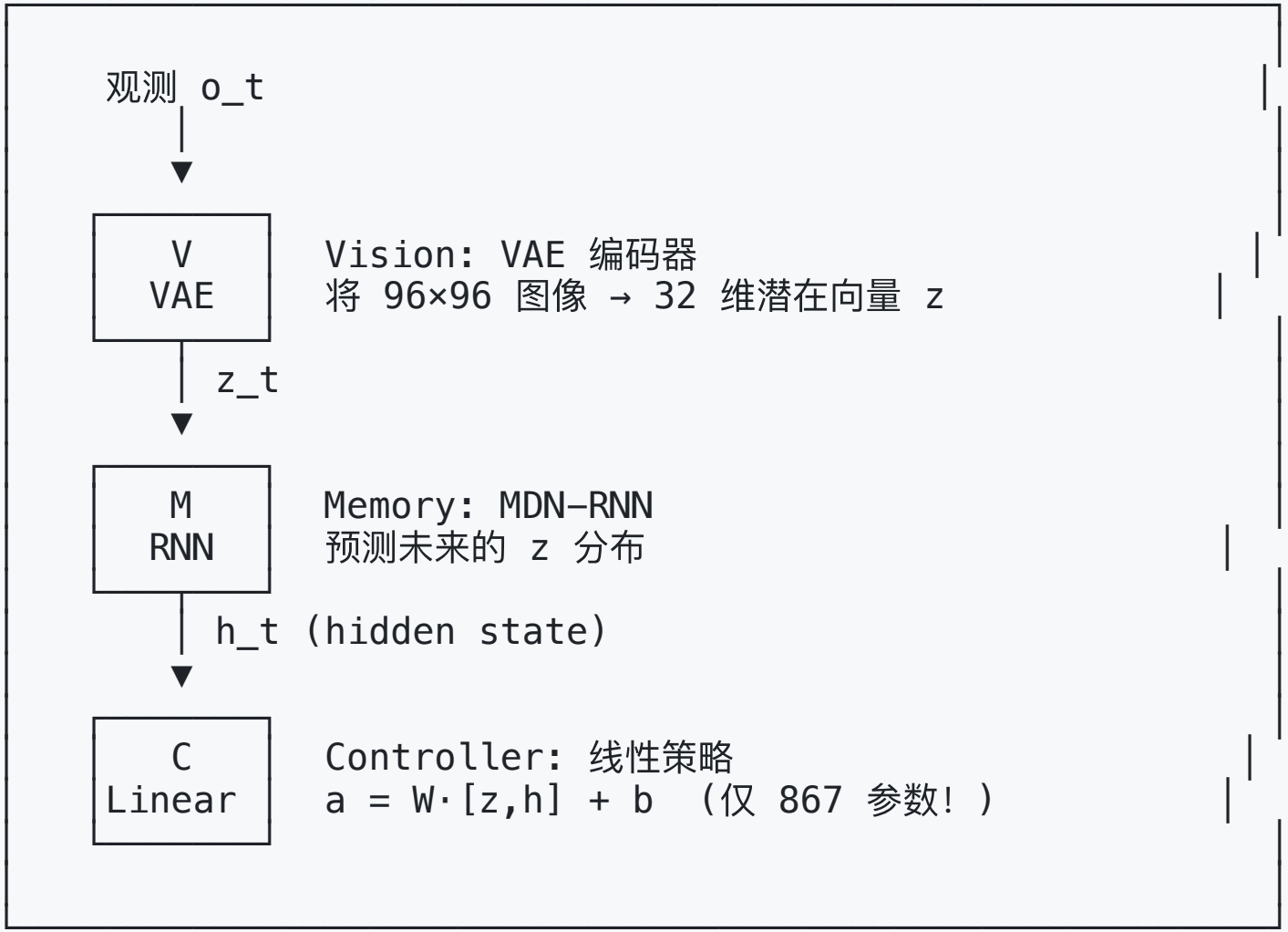
- 预测未来状态
- 评估动作后果
- 支持"心理模拟"

第二部分

🌍 World Models (2018)

Ha & Schmidhuber 的开山之作

核心架构：V-M-C


观测 o_t
V
VAE
Vision: VAE 编码器
将 96×96 图像 → 32 维潜在向量 z
z_t
M
RNN
Memory: MDN-RNN
预测未来的 z 分布
h_t (hidden state)
C
Linear
Controller: 线性策略
a = W·[z,h] + b （仅 867 参数！）

V: VAE 编码器

直觉

把复杂图像"压缩"成简洁表示

```
96×96×3 RGB
     ↓ 编码
32 维向量 z
     ↓ 解码
96×96×3 RGB
```

数学

编码器： $q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$

解码器： $p_\theta(x|z)$

损失：

$$\mathcal{L} = \underbrace{||x - \hat{x}||^2}_{\text{重建}} + \underbrace{D_{KL}(q||p)}_{\text{正则化}}$$

M: MDN-RNN 动态模型

直觉

学习"世界运行规律"

```
过去的 z → 预测未来的 z

z_1, a_1 → z_2
z_2, a_2 → z_3
...
```

数学

Mixture Density Network:

$$p(z_{t+1}) = \sum_{i=1}^{K} \pi_i \mathcal{N}(\mu_i, \sigma_i)$$

- $K = 5$ 个高斯分量
- 建模多模态未来
- 例：球可能弹左或弹右

C: 线性控制器 + CMA-ES

为什么用线性?

```
a = W · [z, h] + b

参数量:
- W: 3 × (32+256) = 864
- b: 3
- 总计: 867 参数
```

参数少 → CMA-ES 可优化

CMA-ES 优化

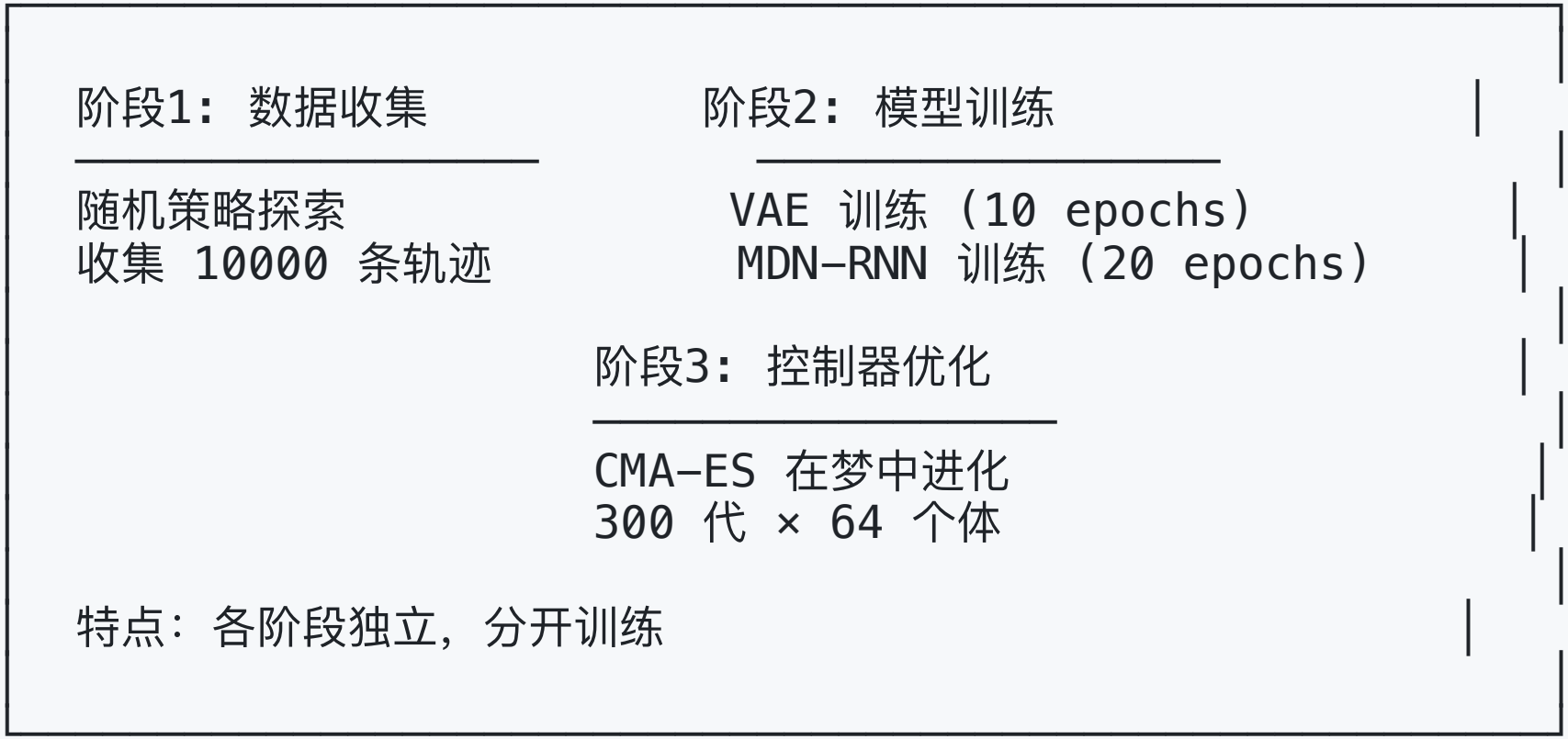
```python
for gen in range(300):
    # 采样 64 个控制器
    population = cma.ask()

    # 在梦中评估
    fitness = [dream_eval(p)
               for p in population]

    # 进化
    cma.tell(population, fitness)
```

无梯度，只看最终奖励

训练流程


阶段1: 数据收集
随机策略探索
收集 10000 条轨迹
阶段2: 模型训练
VAE 训练 (10 epochs)
MDN-RNN 训练 (20 epochs)
阶段3: 控制器优化
CMA-ES 在梦中进化
300 代 × 64 个体
特点: 各阶段独立，分开训练

实验结果：CarRacing

性能

方法	分数
随机策略	~0
DQN	~343
World Models	**~906**
人类	~800

> 证明了"在梦中学习"的可行性

关键发现

- 在梦中训练有效！
- 仅 867 参数的控制器
- 模型能"想象"赛道

World Models 的局限

问题	描述
分阶段训练	V→M→C 无法联合优化
简单控制器	线性策略表达能力有限
CMA-ES 瓶颈	参数多时效率骤降
固定世界模型	训练后无法适应变化

> 需要更强大的方法...

第三部分

🌙 Dreamer 系列 (2020-2023)

从规划到学习的跨越

演进路线

```
2018: World Models
      │ 问题：分阶段、线性控制器
      ▼
2019: PlaNet
      │ 改进：RSSM、CEM 规划
      │ 问题：每步规划太慢
      ▼
2020: Dreamer V1
      │ 改进：Actor-Critic、想象中训练
      │ 问题：连续潜在空间不稳定
      ▼
2021: DreamerV2
      │ 改进：离散潜在空间、KL Balancing
      │ 问题：需要调参
      ▼
2023: DreamerV3
      │ 改进：symlog、固定超参数
      │ 成就：首次解决 Minecraft 钻石
      ▼
```

Dreamer 的核心改进

World Models

分阶段训练
VAE + MDN-RNN
线性 Controller
CMA-ES（无梯度）

Dreamer

端到端联合训练
RSSM（更强）
神经网络 Actor-Critic
策略梯度（有梯度）

RSSM：双路径设计

直觉

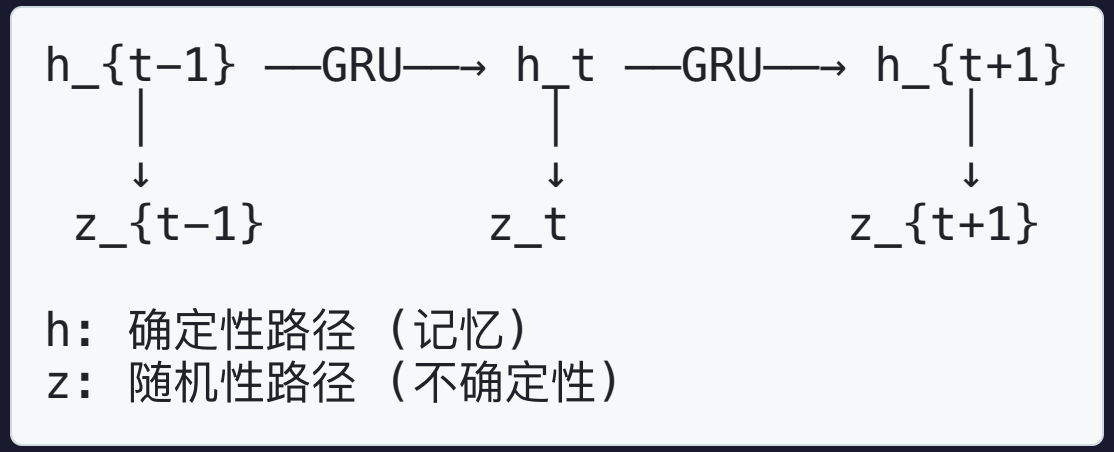
确定性 h: 过去发生的是确定的

- 长期记忆
- 无损传递

随机性 z: 未来是不确定的

- 多种可能
- 建模随机性

架构



RSSM 数学形式

确定性路径：

$$h_t = \mathrm{GRU}_\theta([z_{t-1}, a_{t-1}], h_{t-1})$$

先验分布（想象时用）：

$$p_\theta(z_t | h_t)$$

后验分布（训练时用）：

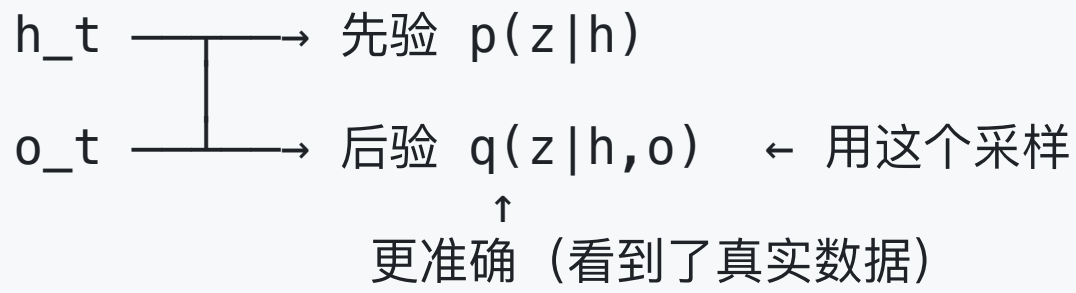
$$q_\phi(z_t | h_t, o_t)$$

状态特征：

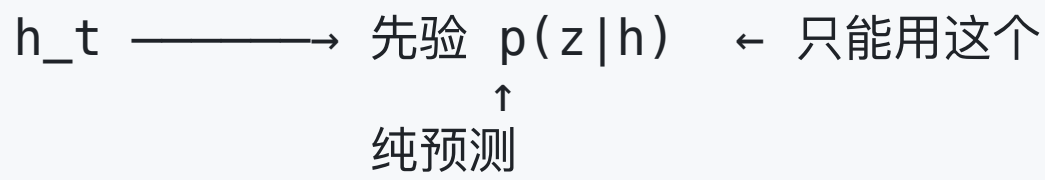
$$s_t = [h_t, z_t]$$

先验 vs 后验

训练时：有真实观测 o_t

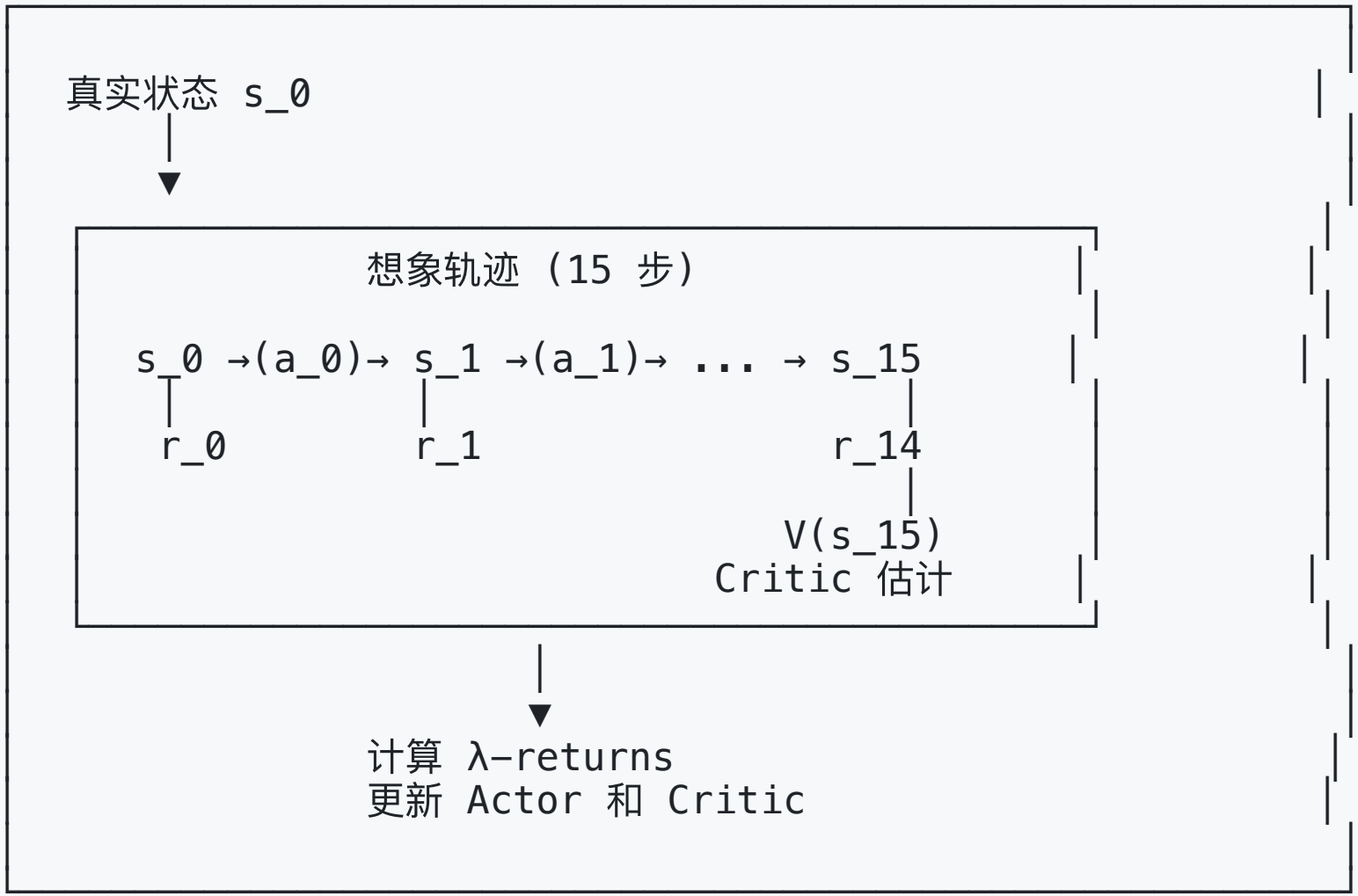


想象时：没有观测



KL 散度让两者尽量一致！

在想象中训练 Actor-Critic



为什么只想象 15 步?

有梯度 (Dreamer)

梯度误差会累积!

$\varepsilon_1 \rightarrow \varepsilon_2 \rightarrow ... \rightarrow \varepsilon_{15}$
可能指数放大

需要控制步数
但有 Critic 估计剩余价值

无梯度 (CMA-ES)

只看最终奖励

$\varepsilon_1 + \varepsilon_2 + ... + \varepsilon_{1000}$
被平均掉

可以跑很长
但需要大量采样

类比:

- Dreamer = 拿精确指南针走 15 步
- CMA-ES = 蒙眼走 1000 步凭感觉

λ-Returns：平衡信任

$$V_t^\lambda = r_t + \gamma \left[(1-\lambda) V(s_{t+1}) + \lambda V_{t+1}^\lambda \right]$$

λ 值	含义	特点
λ=0	完全信任 Critic	高偏差，低方差
λ=1	完全用真实奖励	低偏差，高方差
λ=0.95	Dreamer 默认	**平衡**

> 短期用真实奖励，长期靠 Critic 估计

DreamerV2：离散潜在空间

为什么离散?

连续 $z \sim N(\mu,\sigma^2)$:
- "有敌人" = 0.73 ?
- 边界模糊
- 容易后验坍缩

离散 $z \sim$ Categorical:
- "有敌人" = [1,0]
- 清晰边界
- 训练更稳定

具体设计

$z = [z^1, z^2, ..., z^{32}]$

每个 z^i 是 32 类 one-hot
总组合: $32^{32} \approx \infty$

Straight-Through 梯度:
前向: argmax (离散)
反向: softmax (连续梯度)

Straight-Through Gradient

```
def straight_through(logits):
    probs = softmax(logits)

    # 前向：离散采样
    indices = probs.argmax(dim=-1)
    one_hot = F.one_hot(indices, 32)

    # 反向：假装是连续的
    z = one_hot + probs - probs.detach()
    #   ↑ 常数     ↑ 有梯度   ↑ 无梯度

    return z
```

效果： 前向得到离散值，反向梯度流过 `probs`

KL Balancing

$$\mathcal{L}_{KL} = \alpha \cdot D_{KL}[\text{sg}(q) \| p] + (1 - \alpha) \cdot D_{KL}[q \| \text{sg}(p)]$$

α = 0.8

传统 KL：同时更新 q 和 p
问题：后验 q 可能"躲避"到先验 p 覆盖的区域

DreamerV2：分开更新
- L_dyn：固定 q，让 p 去拟合 q → 先验变强
- L_rep：固定 p，让 q 向 p 靠拢 → 后验规整

为什么偏向让先验变强？
→ 因为想象时只能用先验！

DreamerV3：通用超参数

核心技巧：symlog

$$\text{symlog}(x) = \text{sign}(x) \cdot \ln(|x| + 1)$$

不同任务的奖励尺度：
- Pong: {-1, 0, +1}
- Breakout: [0, 400+]
- Minecraft: [0, 1000+]

symlog 压缩效果：
- symlog(1) = 0.69
- symlog(100) = 4.62
- symlog(10000) = 9.21

同一套超参数适用所有任务！

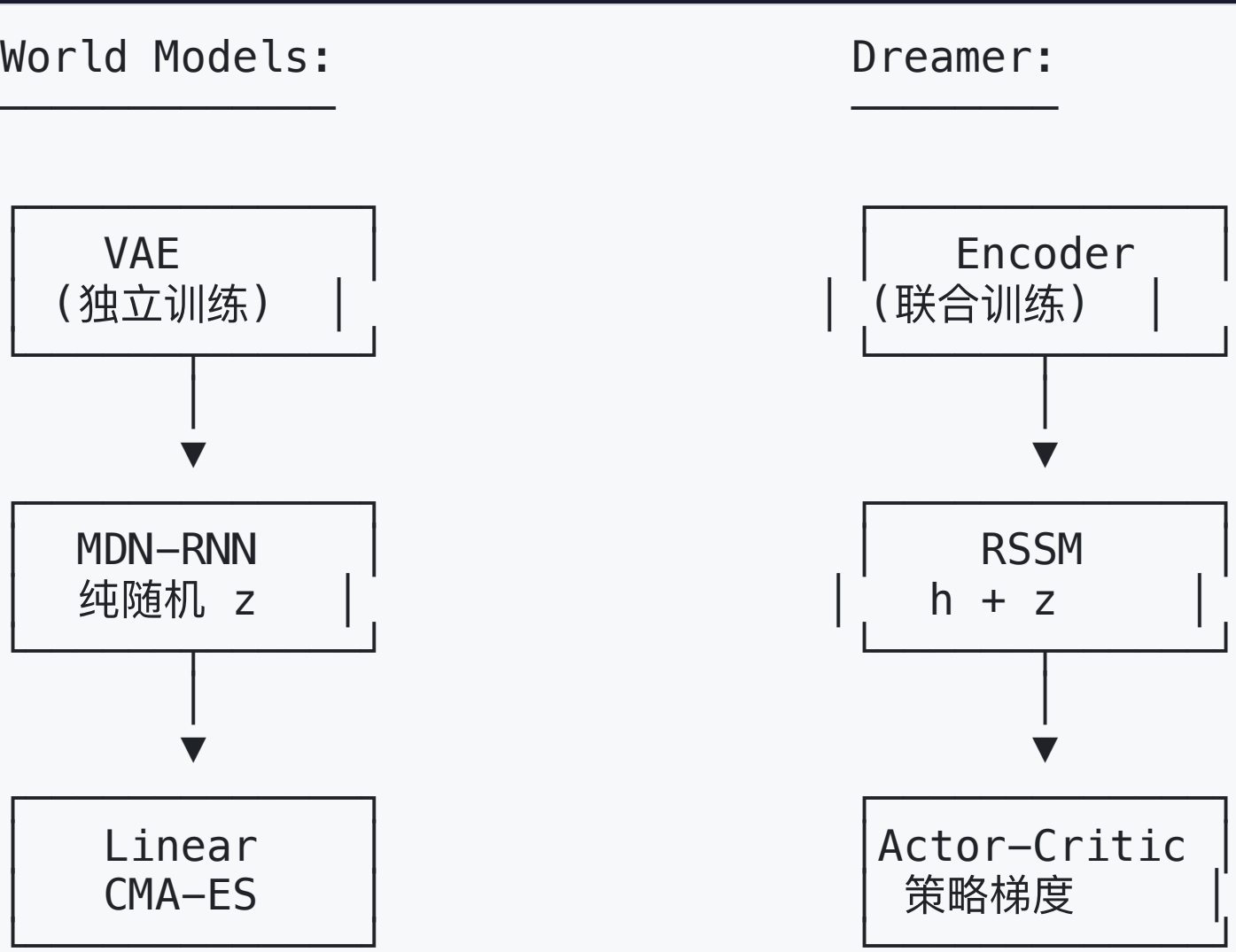
性能对比

任务	World Models	Dreamer V1	V2	V3
CarRacing	~906	-	-	-
Atari 55 游戏	-	115% 人类	**200%**	SOTA
DMControl	-	SOTA	SOTA	SOTA
Minecraft 钻石	-	✗	✗	首次成功 ✓

第四部分

🔬 核心技术深度剖析

对比总结：架构


World Models:
VAE
(独立训练)
MDN-RNN
纯随机 z
Linear
CMA-ES
Dreamer:
Encoder
(联合训练)
RSSM
h + z
Actor-Critic
策略梯度

对比总结：训练

World Models

```
阶段 1: 收集数据
        ↓ (固定)
阶段 2: 训练 V, M
        ↓ (固定)
阶段 3: 进化 C

特点: 流水线
问题: 无法联合优化
```

Dreamer

```
while training:
    # 交替进行
    1. 真实环境交互
    2. 更新世界模型
    3. 想象 + 更新策略

特点: 持续迭代
优势: 联合优化
```

对比总结：关键设计

维度	World Models	Dreamer
状态表示	纯随机 z	确定 h + 随机 z
动态模型	MDN (混合高斯)	先验 + 后验
潜在空间	连续	离散 (V2+)
策略优化	CMA-ES	策略梯度
价值估计	无	Critic 网络
想象长度	~1000 步	~15 步

三阶段割裂问题

问题：三个目标不完全一致！

例子：
- 世界模型完美重建背景草地纹理
- 但这对"躲避敌人"毫无帮助

理想：世界模型应该学习"对决策有用"的信息

第五部分

🚀 新一代方向

方向 1：Genie (2024)

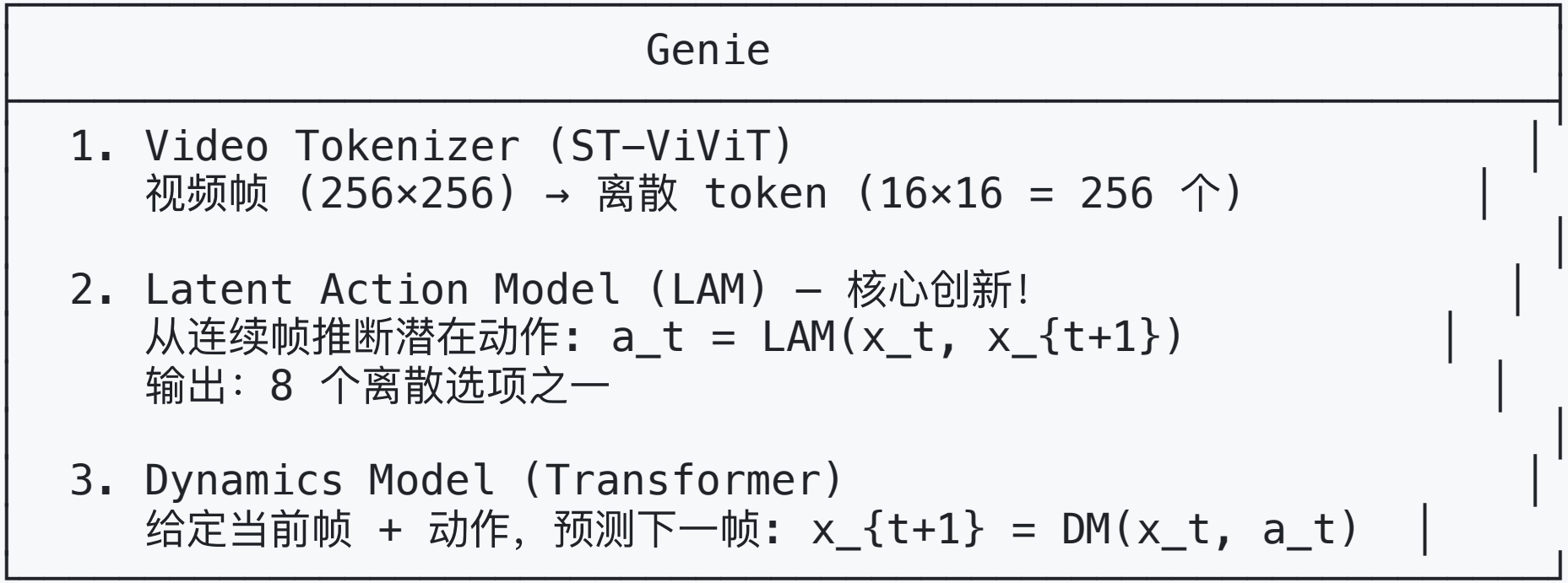
核心问题：如何从无标签视频学习？

传统世界模型需要：（观测 o_t，动作 a_t，奖励 r_t）
必须有明确的动作标签！

互联网上的视频：海量游戏视频、教学视频
只有画面，没有动作标签
玩家按了什么键？不知道！

Genie 的问题：能否从纯视频中，自动发现"动作"是什么？

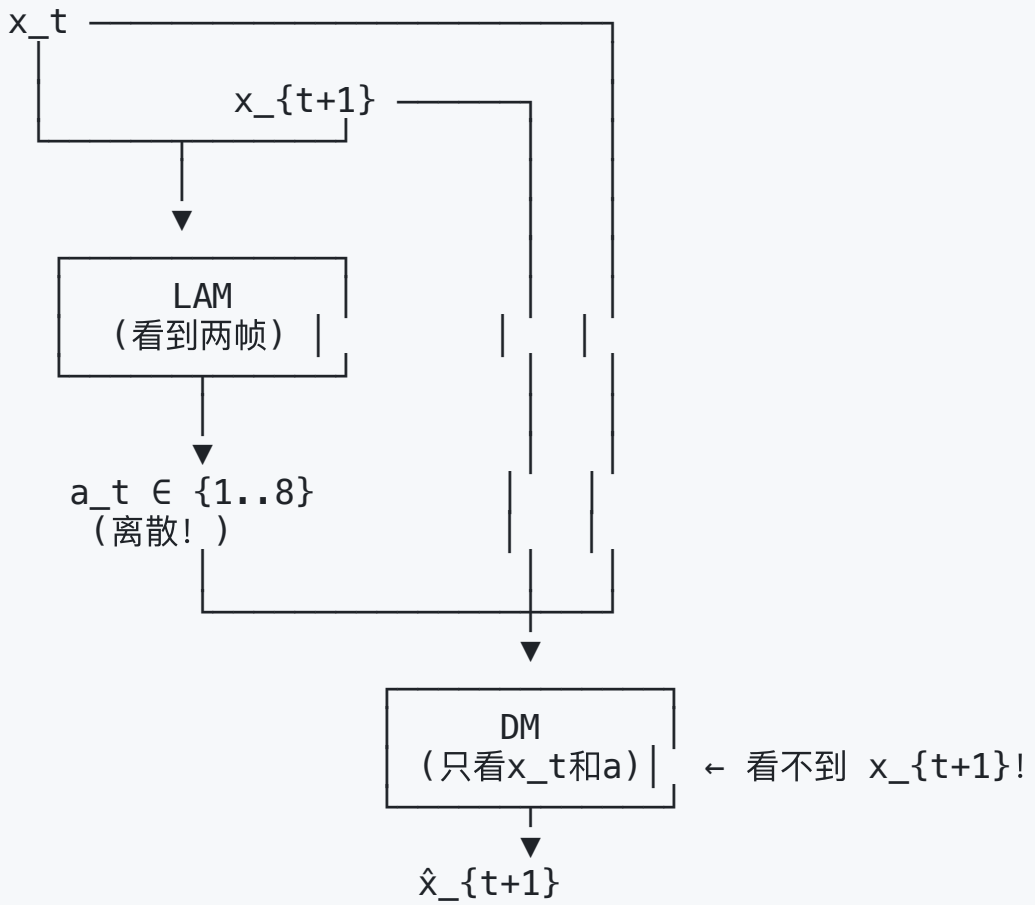
Genie 架构：三大组件



LAM 的信息瓶颈机制

"鸡和蛋"问题：LAM 和 DM 如何联合训练？

答案：信息不对称 + 离散瓶颈



为什么必须是离散瓶颈?

如果 a_t 是连续向量?

问题：信息泄露

LAM 可以直接把 x_{t+1}
的信息编码进 a_t

极端情况：
a_t = Encoder(x_{t+1})
x̂ = Decoder(a_t)
Loss = 0，完美!
但 a_t 毫无语义

离散瓶颈 (8 选 1)

a_t 只有 3 bit 信息
LAM 无法把像素细节塞进去
只能传递"最本质的变化"

结果：
  a_t = 1 → "角色向左"
  a_t = 2 → "角色向右"
  a_t = 3 → "角色跳跃"
  ...

关键 Trade-off：不需要动作标签，但动作空间被限制为 K 个离散选项

Genie 的效果与局限

获得：
+ 可以用海量无标签视频（YouTube、网络视频）
+ 自动发现"有意义"的动作
+ 从草图生成可玩游戏

付出：
- 动作空间被限制为 K 个离散选项
- K 太小 → 表达力不足（无法表达复杂动作）
- K 太大 → 回到信息泄露问题

本质：用"动作空间的先验假设"换"不需要标签"

方向 2：LeCun 的 JEPA

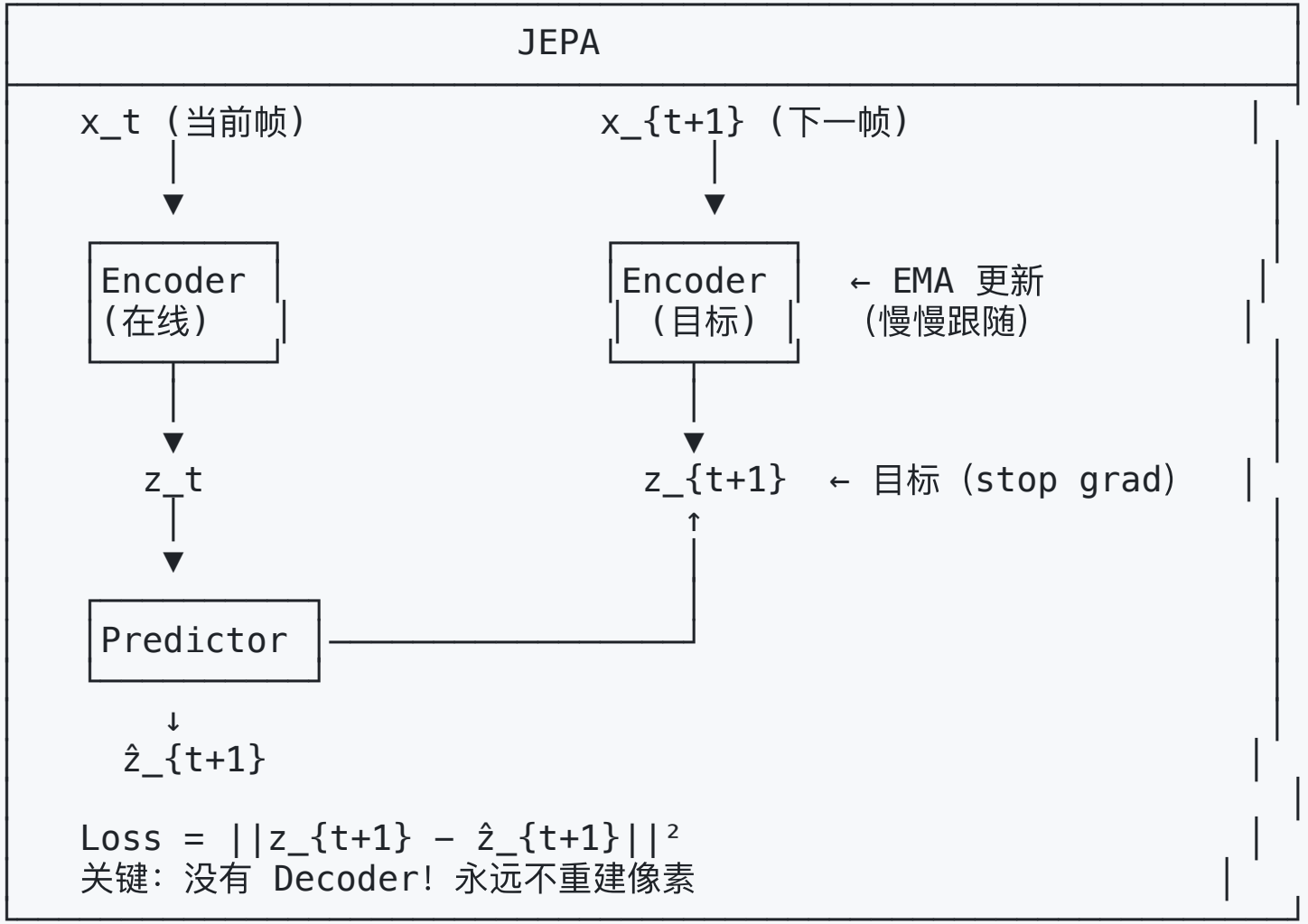
LeCun 对生成式方法的批评

```
"你们都在预测像素（或像素的压缩表示）"
"为什么要预测草地上每片叶子的位置？"
"这些细节对决策有用吗？"

生成式 (Dreamer/Genie):
  x_t → Encoder → z_t → Decoder → x̂_{t+1}
  必须预测所有像素细节

JEPA (判别式):
  x_t → Encoder → z_t → Predictor → ẑ_{t+1}
  x_{t+1} → Encoder → z_{t+1}  (目标)
  只预测"重要的"抽象特征，永远不碰像素！
```

JEPA 架构与 EMA


JEPA
x_t（当前帧）
x_{t+1}（下一帧）
Encoder
（在线）
Encoder
（目标）
← EMA 更新
（慢慢跟随）
z_t
z_{t+1} ← 目标（stop grad）
Predictor
ẑ_{t+1}
Loss = ||z_{t+1} − ẑ_{t+1}||²
关键：没有 Decoder！永远不重建像素

EMA 防止表示坍缩

问题：常数输出

如果 Encoder 学到：
$z_t = [0, 0, 0, ...]$
$z_{t+1} = [0, 0, 0, ...]$
$\hat{z}_{t+1} = [0, 0, 0, ...]$

Loss = 0，完美！
但表示毫无信息
→ 表示坍缩

EMA 解决方案

在线 Encoder：正常训练
目标 Encoder：EMA 更新

$$\theta_{target} = 0.99 \times \theta_{target} + 0.01 \times \theta_{online}$$

类比：
领跑者停下 → 跟随者还在追
→ 领跑者必须继续跑

JEPA 的关键优势

vs Dreamer：不需要假设分布！

Dreamer:
  先验 p(z|h) = Categorical(32×32) ← 必须指定分布形式
  后验 q(z|h,o) = Categorical(32×32)
  用 KL 散度约束，需要调节 KL 权重、free bits 等

JEPA:
  z_t = Encoder(x_t) ← 就是一个向量，不是分布
  用 MSE 直接比较，简单直接
  不假设任何分布！

另一个特点：JEPA 不建模不确定性
  Dreamer: p(z_{t+1}) 是分布，可以采样多种可能
  JEPA: $\hat{z}_{t+1}$ 是确定性预测

  LeCun 的观点：低层不确定性（像素级）对决策无关

JEPA vs 生成式

方面	生成式 (Dreamer)	JEPA
预测目标	像素/token	表示
解码器	需要	不需要
分布假设	需要（高斯/离散）	不需要
不确定性	显式建模	不建模
计算效率	较低	较高
RL 应用	成熟	探索中

> LeCun: "预测像素是错误的方向"

方向 3：统一 Transformer 架构

```
输入序列: [obs_1, act_1, rew_1, obs_2, act_2, ...]

                ↓ Transformer ↓

输出: 预测下一个 token

代表工作:
- Decision Transformer: RL 变成序列建模
- Gato: 一个模型做所有任务
- RT-2: 视觉-语言-动作统一
```

Decision Transformer 深度解析

核心思想：把 RL 变成序列建模

```
传统 RL:
  学习 V(s) 或 Q(s,a) → 用价值函数优化策略
  依赖 Bellman 方程: V(s) = r + γV(s')

Decision Transformer:
  把轨迹看成序列: τ = (R̂₁, s₁, a₁, R̂₂, s₂, a₂, ...)
  学习条件生成: P(aₜ | R̂ₜ, sₜ, 历史)

  R̂ₜ = Return-to-Go = 从 t 到结束的累积奖励
```

关键洞察：不预测"最优动作"，而是"想要 X 分时该怎么做"

Credit Assignment: TD vs DT

类比：RNN vs Transformer 的信息传播

为什么 DT 避开了 RL 的"致命三角"?

RL 的致命三角 (Deadly Triad):

1. Function Approximation (神经网络估计)
2. Bootstrapping (用估计值更新估计值)
3. Off-policy Learning (学习非当前策略的数据)

三者同时存在 → 训练不稳定、发散

Decision Transformer 如何绕过?

✗ 不需要 Bootstrapping
- TD: $V(s) \leftarrow r + \gamma \cdot \hat{V}(s')$ ← 用估计更新估计
- DT: 直接用真实的 $\hat{R}_t$, 不估计任何价值函数

✗ 不需要 Value Function
- 不学 V(s), 不学 Q(s,a)
- 只学 $P(a \mid \hat{R}, s)$, 纯监督学习

✓ 可以 Off-policy
- 在任意历史数据上训练
- 推理时指定目标 $\hat{R}$ 生成动作

DT vs TD: 稀疏奖励场景

任务: 100 步迷宫，只有终点 +1 奖励

TD Learning 的困境:
  步骤 1–99: r = 0
  步骤 100: r = 1

  $V(s_{99}) \leftarrow 0 + \gamma \cdot 1 = \gamma$ 第 1 轮学到
  $V(s_{98}) \leftarrow 0 + \gamma^2 = \gamma^2$ 第 2 轮学到
  ...
  $V(s_1) \leftarrow \gamma^{99}$ 第 99 轮才学到!

  + 每轮估计误差累积...

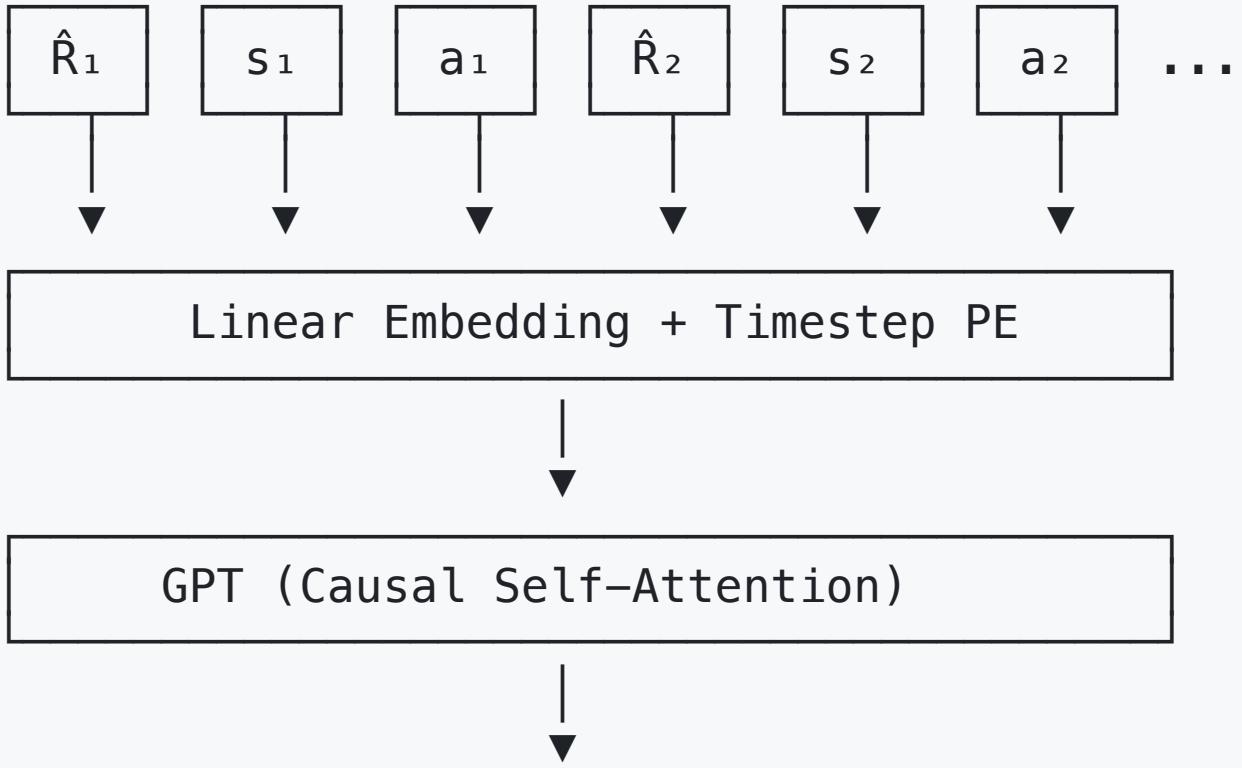
Decision Transformer:
  训练数据: $(\hat{R}=1, s_1, a_1), (\hat{R}=1, s_2, a_2), ...$

  Self-Attention 直接学到:
    "$\hat{R}=1$ 时在 s_1 执行 a_1"
    "$\hat{R}=1$ 时在 s_{50} 执行 a_{50}"

  一次训练，全局 credit assignment!

DT 的架构细节

输入序列（交错排列）：




```
训练: Loss = MSE(a_pred, a_true)  # 纯监督!
推理: 设定 R̂₁ = 目标回报, 自回归生成动作
```

DT 的局限性

1. 依赖数据质量
   - 只能生成"数据中见过的"行为
   - 无法超越数据集中的最优轨迹
   - vs Dreamer: 可以在想象中探索新策略

2. 不学习世界模型
   - 不理解"世界如何运转"
   - 只学"好轨迹长什么样"
   - 无法做 planning 或 model-based 推理

3. 离线 RL 为主
   - 需要大量离线数据
   - 在线学习场景不如传统 RL

4. Return 条件化的挑战
   - 推理时需要设定合理的目标 $\hat{R}$
   - 设太高 → 生成不出来的动作
   - 设太低 → 表现不佳

World Models vs Decision Transformer

维度	World Models/Dreamer	Decision Transformer
核心思想	学习世界如何运转	学习好轨迹长什么样
学习目标	状态转移 $P(s' \mid s,a)$	动作生成 $P(a \mid \hat{R},s)$
是否学世界模型	是	否
能否 Planning	是（想象中规划）	否
能否超越数据	是（想象中探索）	否（受限于数据）
Credit Assignment	Bellman backup	Self-Attention
稀疏奖励	困难	自然处理
在线学习	强	弱

本质区别：

- World Models: "理解世界" → 想象 → 决策

三方对比：哲学分歧

三种方法都在回答同一个问题：如何让 AI 理解世界的运行规律？

Dreamer: "给我标注好的交互数据，我学完整的世界动态"
需要动作标签，但动作空间灵活

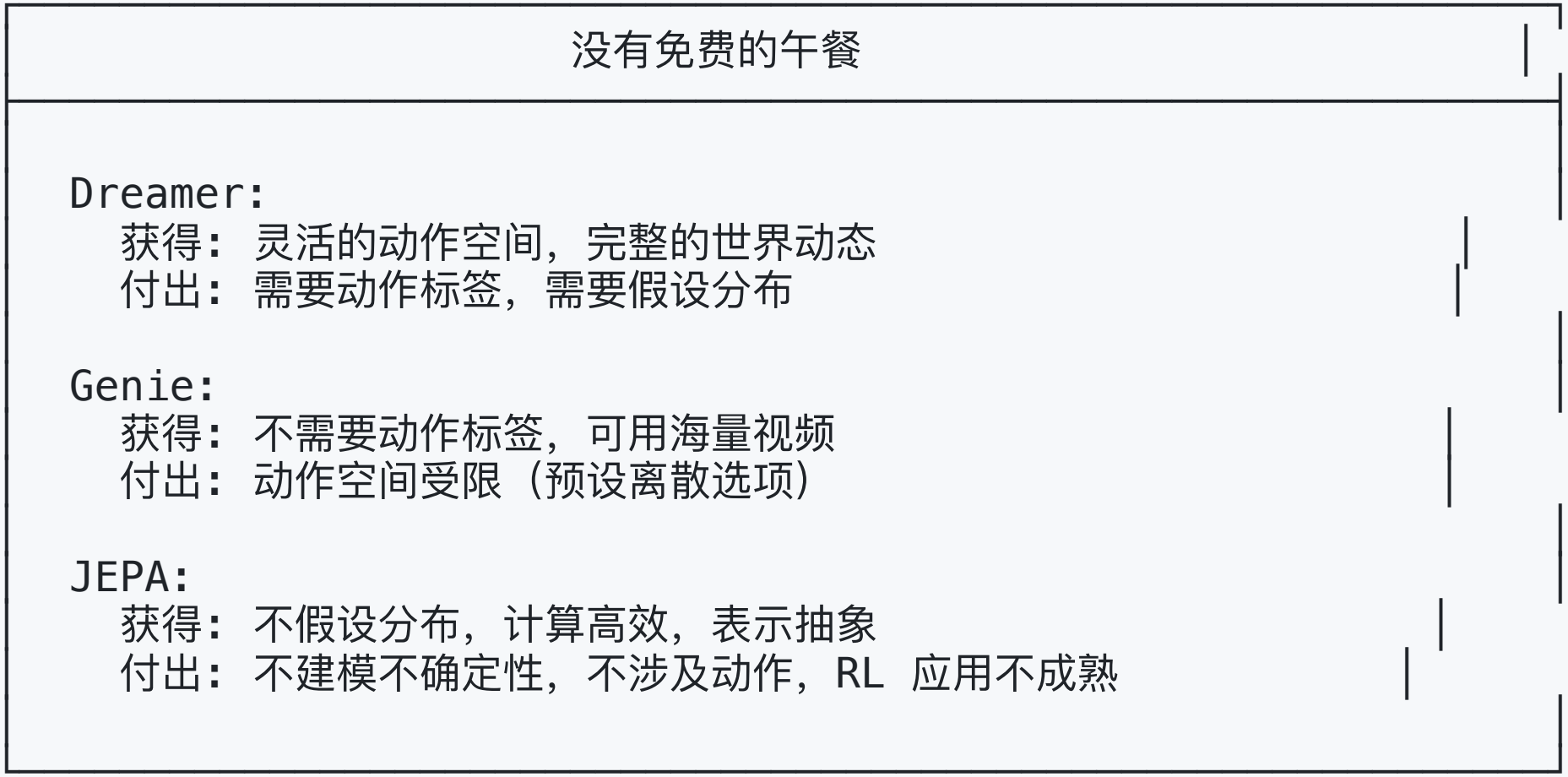
Genie: "只给视频就行，但我只能学有限种动作"
不需要标签，但动作空间受限（8 个离散选项）

JEPA: "我不关心动作，只学习状态的抽象表示"
不涉及动作，专注于表示学习

三方对比：功能表格

维度	Dreamer	Genie	JEPA
预测目标	潜在状态分布 + 像素	像素 token	抽象表示
需要解码器	是	是	否
需要动作标签	是	否	N/A
动作空间	环境定义（灵活）	预设离散	N/A
分布假设	需要	不需要	不需要
建模不确定性	显式 (KL)	隐式	不建模
RL 应用成熟度	高	中	低

核心 Trade-off：没有免费的午餐



适用场景选择

选择 Dreamer

- 有交互环境（模拟器、游戏）
- 需要精确控制
- 追求 RL 性能
- 例：Atari、机器人、DMControl

选择 Genie

- 有大量无标签视频
- 动作空间相对简单
- 想从视频创建可交互环境

选择 JEPA

- 主要目标是学习好的表示
- 下游任务是分类、检测等
- 追求计算效率
- 例：视觉预训练、视频理解

未来融合?

- Dreamer 的 RL 能力
- Genie 的无监督动作发现
- JEPA 的抽象表示

未来展望

```
第一代 (2018): World Models
  证明"在梦中学习"可行

第二代 (2020–2023): Dreamer 系列
  端到端、强大性能

第三代 (2024+): ???
  ├── 大规模视频预训练 (Genie)
  ├── 抽象表示预测 (JEPA)
  ├── 语言集成 (RT-2)
  └── 因果推理
```

第六部分

📝 总结

核心洞察

1. 确定性 + 随机性分离

```
h 负责记忆，z 负责随机性
类比：Transformer 的残差连接也是"确定性路径"
```

2. 短视野 + 价值估计 > 长视野

```
有了 Critic，不需要想象太远
V(s) 压缩了"未来所有信息"
```

3. 离散表示更稳定

```
连续空间容易"漂移"和"坍缩"
离散空间有明确的边界
```

开放问题

1. **什么是"好的"世界模型?**
   重建准确? 决策有用? 泛化能力?
2. **三阶段如何统一?**
   观察-建模-决策 如何端到端为决策服务?
3. **与大语言模型的关系?**
   LLM 是否已经是某种世界模型?
4. **从模拟到现实**
   如何跨越 Sim2Real 的鸿沟?

实践建议

用 World Models

- 简单任务
- 快速原型
- 需要可解释性
- 计算资源有限

用 Dreamer

- 复杂任务
- 追求性能
- 持续学习
- 有足够算力

资源

论文：

- World Models (2018): arxiv.org/abs/1803.10122
- Dreamer (2020): arxiv.org/abs/1912.01603
- DreamerV3 (2023): arxiv.org/abs/2301.04104
- Genie (2024): arxiv.org/abs/2402.15391

代码：

- 官方 DreamerV3: github.com/danijar/dreamerv3
- PyTorch 复现: github.com/NM512/dreamerv3-torch

🙏 谢谢！

Q&A

附录：核心公式速查

组件	公式
VAE ELBO	$\log p(x) \geq \mathbb{E}_{q}[\log p(x
RSSM 确定性	$h_t = \mathrm{GRU}([z_{t-1}, a_{t-1}], h_{t-1})$
RSSM 先验	$p(z_t \mid h_t) = \prod_i \mathrm{Cat}(\pi^i(h_t))$
λ-Returns	$V_t^\lambda = r_t + \gamma[(1-\lambda)V(s_{t+1}) + \lambda V_{t+1}^\lambda]$
KL Balance	$\alpha \cdot D_{KL}[\mathrm{sg}(q) \mid p] + (1-\alpha) \cdot D_{KL}[q \mid \mathrm{sg}(p)]$
Symlog	$\text{symlog}(x) = \text{sign}(x) \cdot \ln(

附录：演进时间线

```
2018.03  World Models 发布
2019.02  PlaNet 引入 RSSM
2020.01  Dreamer V1 Actor-Critic
2021.01  DreamerV2 离散潜在空间，Atari 超越人类
2022.06  LeCun 发布 JEPA 路线图
2022.11  Gato 通用智能体
2023.01  DreamerV3 通用超参数，解决 Minecraft
2023.04  I-JEPA 图像自监督
2024.02  Genie 从视频学习可控世界
2024.02  V-JEPA 视频自监督
```