

2025 年 10 月 20 日至 10 月 26 日周报

何瑞杰
中山大学, 大湾区大学

1. 项目进展

1.1. 使用神经网络学习生命游戏的演化动力学

1.1.1. 部分模型的权重可视化

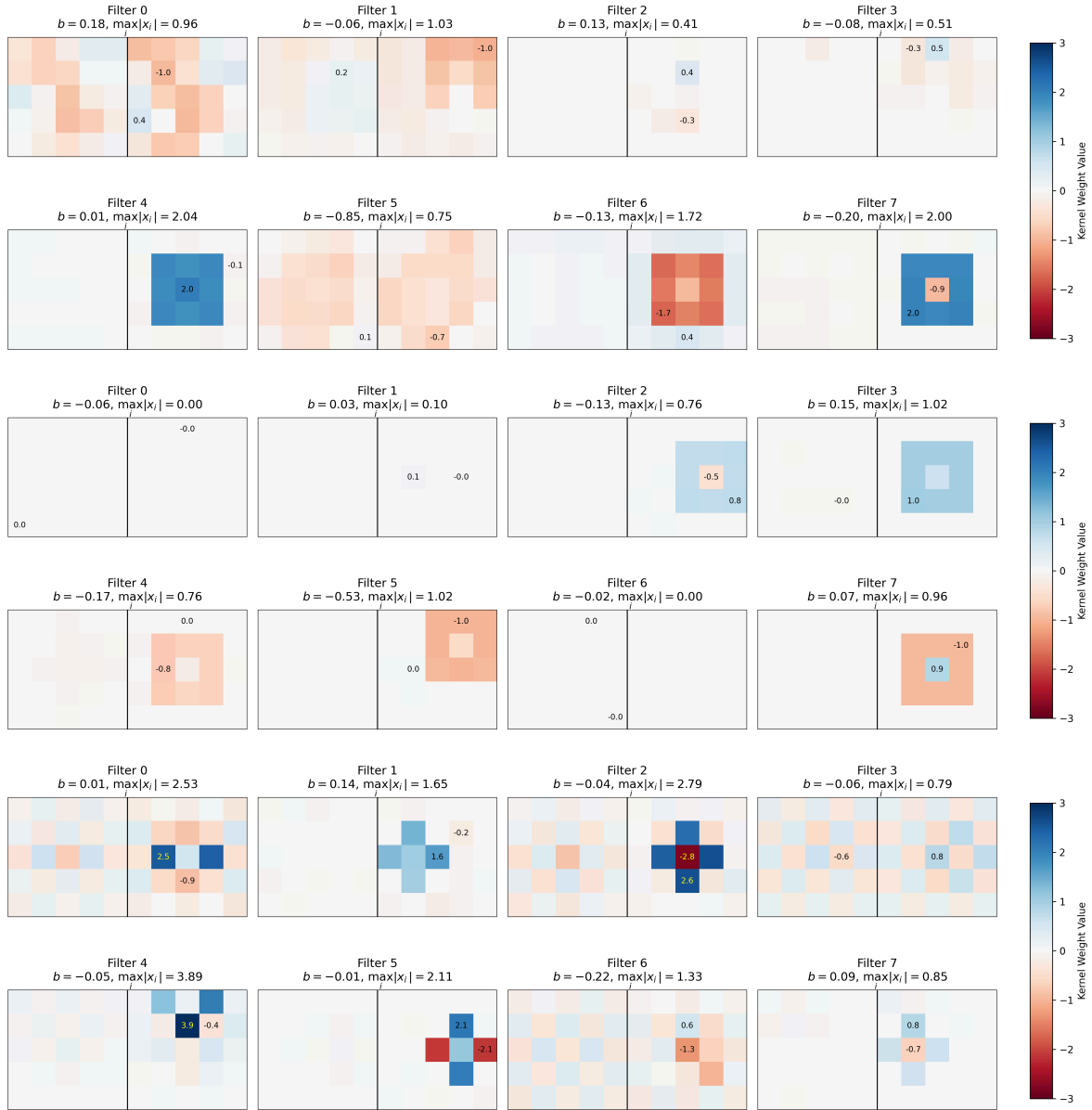


Figure 1: 不同规则下训练模型 CNN-Small 的首层卷积核可视化。

从上至下的规则依次为: B3/S23, B2/S, B13/S012V

Figure 1 展示了在不同规则的训练集上训练得到的 CNN-Small 模型的第一层卷积核热力图。图中从上至下包括在以 B3/S23、B2/S 和 B13/S012V 为演化规则之训练集上得到的结果。其中前

两个规则的邻域为 Moore 邻域，最后一个为 Von-Neumann 邻域。可以看到，上图中的 4, 6, 7 号卷积核、中图中的 2, 3, 4, 5, 7 号卷积核以及下图中的 1, 2, 5, 7 都清晰地显示出了对应邻域种类的模式。

我们期望通过网络中一定的稀疏化约束，从而使得卷积网络的首层权重仅出现其所训练的数据集之规则的邻域模式。然而我们注意到，B3/S23 训练得到的卷积核和 B13/S012V 训练得到的卷积核中，除了清晰的邻域模式之外，依然存在较多的噪音，这要求我们提高稀疏化约束的权重。有趣的是，B2/S23 模型的第 4, 6, 7 号卷积核非常接近旋转对称性，B2/S 的第 5 号卷积核以及 B13/S012V 的第 1, 4, 6, 7 号卷积核的邻域模式并未出现在卷积核的中央。此外，B13/S012V 的首层卷积核呈现了正负交替的棋盘格模式。

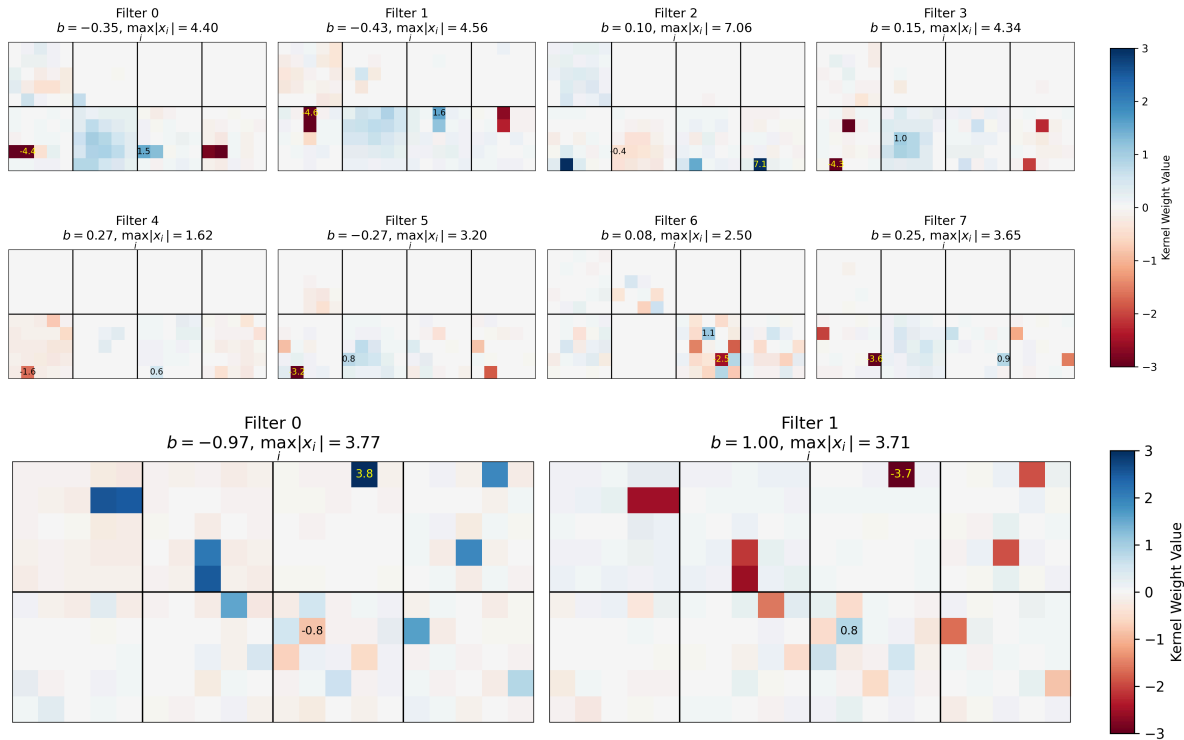


Figure 2: B3/S23 规则下训练模型 CNN-Small 的卷积核可视化。上图为第二层，下图为第三层。

在第二层和第三层，权重稀疏化起到了较好的效果，以 B3/S23 训练的模型为例。如 Figure 2 所示，模型的第二层和第三层卷积层的数量很少，对于权重几乎为零的卷积层，在之后做分析时，可以恰当忽略；对于卷积核中较低的权重，我们也可以忽略。

值得注意的是，现实的训练结果和我们的理想存在差距：如上文中提到的邻域模式不出现在卷积核中央的问题，它的一个可能解释是，由于第 2 层和第 3 层中的卷积核具有足够灵活性，在训练中也或许训练得到了适应邻域特征偏置的卷积核，即第一层学习到的卷积核是 $L_t \circ g$ ，其中 L_t 是某个平移变换，接下来的层中学到的可能是 $f \circ L_{t-1}$ ，这样就能实现在组合后其效果等价于 $f \circ g$ 。

1.1.2. 神经网络作为演化模拟器的效果验证

对 CNN-Small 作为演化器的一步迭代中中心元胞的状态转换和其邻居存活细胞数的统计结果如 Figure 3 所示。左上方为在同样规则数据上的测试结果，右上方为在 B3/S23 规则数据上的测试结果，下方为 B345/S5 数据本身的统计结果。对比左上方和下方，可以看到模型有一定的外推能力，在原始数据中，dead \rightarrow living 的数据严重不平衡，但模型仍然学到了正确的演化规

律。与此同时, living \rightarrow living 也即存活的情况下, 模型偶尔犯错误, 推测是因为对应数据中这样的转化所占比例太小导致。

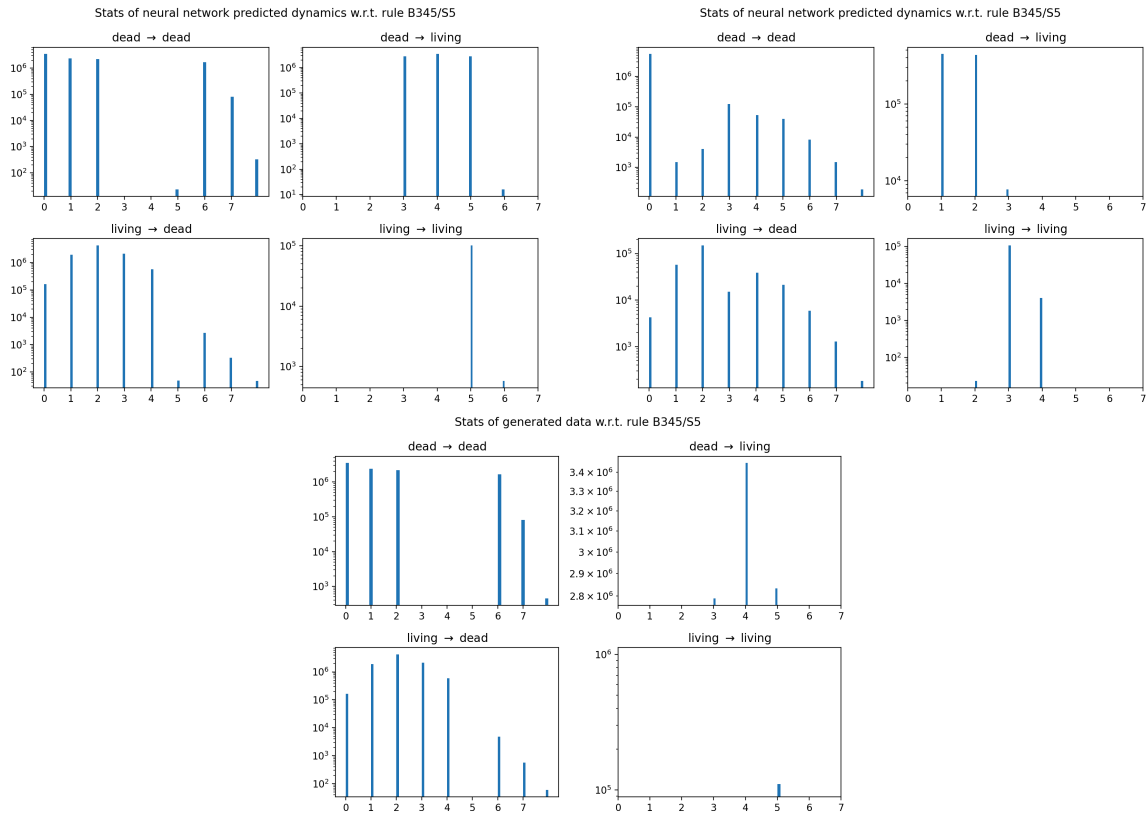


Figure 3: 在 B345/S5 规则数据上训练的 CNN-small 模型在测试数据上的演化模拟统计

另一个有趣的现象是, 当模型接受 B3/S23 的演化轨迹中的状态时, 统计得到的结果和 B3/S23 完全不同, 在诞生和存活的两种情况下甚至其模式和 B3/S23 类似。我们需要进一步调整数据生成方案, 和网络结构, 使得模型即使接受来自其他演化轨道中的状态, 也可以恪守其训练集中的演化规则。

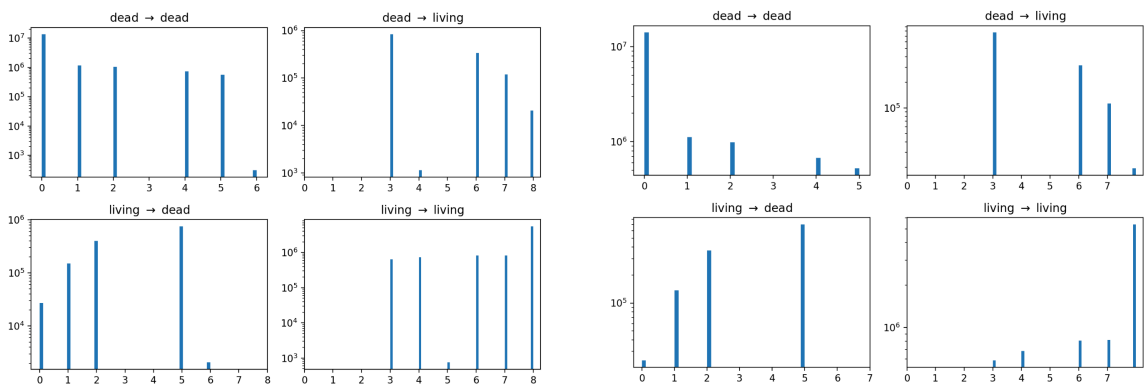


Figure 6: 在 B3678/S34678 规则数据上训练的 CNN-small 模型在测试数据上的演化模拟统计

另外, Figure 6 显示出在如 B3678/S35678 这样的规则中, 由于数据生成式易于陷入不动点或周期点, 尤其是全死亡或全存活的平凡不动点时, 训练数据将带有极大的偏置 (bias), 与此同时神经网络并不能有效学习到正确演化规则的难度因此上升。

1.1.3. 下一步

目前将目标从提取显式的规则改为从已知部分规则和黑箱规则代理产生的少量数据, 结合带有等变性的神经网络训练作为动力学仿真器。规则发现的具体流程是, 从一些已知规则出发,

首先根据一条有偏的轨道从一个规则空间中选取一系列的假设规则，在神经网络学习这条轨道后，再通过神经网络作为演化模拟器的演化统计结果，和原来的统计数据对比，以确定某些候选规则存在或不存在，从而逐步确定系统演化的真正动力学。

2. 文献阅读

2.1. Score-based generative modeling through SDE [1]

Yang Song et al. | <https://arxiv.org/abs/2011.13456>

本文从 SDE 的视角统一了先前的 Langevin 退火的 score matching 方法 SMLD[2]和原版扩散模型 DDPM[3], 从它们的共同点出发, 先将加噪动力学连续化变成 SDE, 再利用已有的逆向 SDE 解析解, 得到与加噪 SDE 相反演化方向的逆向 SDE; 最后从 SMLD 和 DDPM 逆向求解的数值算法提出了新的预测-校正方法和概率流 ODE 方法。可以说本文所提出的是随机采样的改进, 神经网络扮演的角色仅仅为 $\nabla \log p(\mathbf{x})$ 的逼近器, 不是本文的重点。

2.1.1. Score matching: SMLD 和 DDPM

Score-based 生成模型的核心思想是通过学习 $\mathbf{s}_\theta(\mathbf{x}, t)$, 让其逼近 $\nabla \log p(\mathbf{x})$, 然后再根据 Langevin 采样, 生成来自未知分布 $p(\mathbf{x})$ 的样本。在 SMLD 中, 由于数据分布未知而无法计算的目标项 $\nabla \log p(\mathbf{x})$ 是通过加噪解决的, 即考虑高斯核 $q_\sigma(\hat{\mathbf{x}}|\mathbf{x}) \sim \mathcal{N}(\mathbf{0}, \sigma \mathbf{I})$, 加噪后变量对应的目标函数可以写成

$$\begin{aligned} J(\theta) &= \frac{1}{2} \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x}), \hat{\mathbf{x}} \sim q_\sigma(\hat{\mathbf{x}}|\mathbf{x})} \left[\|\mathbf{s}_\theta(\hat{\mathbf{x}}, \sigma) - \nabla_{\hat{\mathbf{x}}} \log q_\sigma(\hat{\mathbf{x}}|\mathbf{x})\|_2^2 \right] \\ &= \frac{1}{2} \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x}), \hat{\mathbf{x}} \sim q_\sigma(\hat{\mathbf{x}}|\mathbf{x})} \left[\left\| \mathbf{s}_\theta(\hat{\mathbf{x}}, \sigma) + \frac{\sigma}{\varepsilon} \right\|_2^2 \right] \quad \hat{\mathbf{x}} = \mathbf{x} + \varepsilon, \varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma \mathbf{I}) \end{aligned} \quad (1)$$

可以看出, 采用加噪技巧的 SMLD 中, 分数函数实际上在预测噪声, 这就与 DDPM 建立了联系。即使 DDPM 论文中并未显式地做出 score matching, 但我们也可以认为 DDPM 是在训练一个分数函数。

现在来看二者的去噪动力学。SMLD 使用 Langevin 采样的方法, 对于某个固定的噪声水平 σ_i , 有

$$\mathbf{x}_i^m = \mathbf{x}_{i-1}^m + \alpha_i \mathbf{s}_\theta(\mathbf{x}_{i-1}^m, \sigma_i) + \sqrt{2\alpha_i} \mathbf{z}_i^m, \quad \mathbf{z}_i^m \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), m = 1, \dots, M \quad (2)$$

其中 $i = 1, \dots, N$ 。DDPM 的版本和 SMLD 相似, 它通过两个 Gauss 分布之间 KL 散度的解析解, 可以求出逆向条件分布 $q(\mathbf{x}_{i-1}|\mathbf{x}_i, \mathbf{x}_0)$ 的均值 $\tilde{\mu}_t$ 和方差 $\tilde{\beta}_t$, 最后得到去噪公式

$$\mathbf{x}_{i-1} = \frac{1}{\sqrt{1-\beta_i}}(\mathbf{x}_i - \beta_i \mathbf{s}_\theta(\mathbf{x}_i, \sigma_i)) + \sqrt{\beta_i} \mathbf{z}_i, \quad \mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (3)$$

2.1.2. Score matching 的 SDE 视角

2.1.2.1. 加噪过程

我们考虑将 SMLD 和 DDPM 的加噪动力学连续化为 $d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)dt + g(t)d\mathbf{w}$ 的形式, 其中 \mathbf{w} 是 n 维 Brown 运动, $\mathbf{f}(\mathbf{x}, t)$ 是漂移项, $g(t)$ 是扩散项。对于 SMLD, 有下面的改写:

$$\begin{aligned} \mathbf{x}_i &= \mathbf{x}_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2} \mathbf{z}_i \\ \mathbf{x}(t + \Delta t) - \mathbf{x}(t) &= \sqrt{\sigma(t + \Delta t)^2 - \sigma(t)^2} \mathbf{z}(t) \\ \mathbf{x}(t + \Delta t) - \mathbf{x}(t) &= \sqrt{\frac{d[\sigma(t)]^2}{dt}} \sqrt{\Delta t} \mathbf{z}(t) \quad \text{用 } \frac{d[\sigma(t)]^2}{dt} \text{ 对 } \sigma_i^2 - \sigma_{i-1}^2 \text{ 做一阶近似} \\ d\mathbf{x} &= \sqrt{\frac{d[\sigma(t)]^2}{dt}} d\mathbf{w} \quad d\mathbf{w} \sim (dt)^{\frac{1}{2}} \varepsilon \end{aligned} \quad (4)$$

类似地, 对 DDPM 的加噪过程, 也有类似的改写:

$$\mathbf{x}_i = \sqrt{1 - \beta_i} \mathbf{x}_{i-1} + \sqrt{\beta_i} \mathbf{z}_{i-1} \quad (5)$$

$$\begin{aligned} \mathbf{x}(t + \Delta t) &= \sqrt{1 - \beta(t + \Delta t)\Delta t} \mathbf{x}(t) + \sqrt{\beta(t + \Delta t)\Delta t} \mathbf{z}(t) && \text{令 } \beta(i/N) = \beta_i \\ &&& \text{其中 } 1/N \text{ 是噪声步长} \\ &\approx \left(1 - \frac{1}{2}\beta(t + \Delta t)\Delta t\right) \mathbf{x}(t) + \sqrt{\beta(t + \Delta t)\Delta t} \mathbf{z}(t) && \text{Taylor 展开到一阶} \\ &\approx \mathbf{x}(t) - \frac{1}{2}\beta(t)\mathbf{x}(t)\Delta t + \sqrt{\beta(t)\Delta t} \mathbf{z}(t) \\ d\mathbf{x} &= -\frac{1}{2}\beta(t)\mathbf{x}dt + \sqrt{\beta(t)}d\mathbf{w} && d\mathbf{w} \sim (dt)^{\frac{1}{2}}\varepsilon \end{aligned} \quad (6)$$

如果 DDPM 中的噪声表不是线性变化的, 那将得到一个形式类似, 但 \mathbf{f} 和 g 不同的 SDE。上文中, SMLD 的加噪 SDE 被称为是 variance exploding (VE) 的, 而 DDPM 的加噪 SDE 被称为是 variance preserving (VP) 的。作者还构造出一种 sub-VP SDE, 加噪 SDE 为

$$d\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}dt + \underbrace{\sqrt{\beta(t)\left(1 - e^{-2\int_0^t \beta(s)ds}\right)}}_{\sqrt{\tilde{\beta}(t)}}d\mathbf{w}. \quad (7)$$

其方差被 VP-SDE 的方差控制。

2.1.3. 求解逆向 SDE

Anderson 给出了上节中一般形式之加噪 SDE 的逆向 SDE, 其形式为

$$d\mathbf{x} = [-\mathbf{f}(\mathbf{x}, t) + g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})]dt + g(t)d\bar{\mathbf{w}} \quad (8)$$

其中 $\bar{\mathbf{w}}$ 是一个倒流的 Brown 运动, dt 是倒流的无穷小时间间隔。由于 \mathbf{f} 和 g 已知, 而 $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ 由神经网络估计。剩下的问题就变成了如何用这个逆向 SDE 采样。

下面这几节还没有整理出来。

2.1.3.1. 一般的 SDE 求解器

2.1.3.2. 预测-校正方法

2.1.3.3. 概率流 ODE

2.1.4. 可控生成

2.1.5. 实验结果和讨论

2.1.6. 依然存在的疑惑

- Variance Exploding SDE 和 Variance Preserving SDE 以及 sub-VP SDE 为什么叫这些名字, 换言之, 如何证明它们分别是 variance exploding 和 variance preserving 的?
- 概率流 ODE 是否只是一个求解逆向 SDE 的方法? 它和文中提出的 SDE 求解方法之间有什么不同?

3. 学习进度

3.1. 随机过程

本周稍稍总结功率谱密度后的故事。

接着我跳过 Poisson 过程，开始看 Markov 链的部分。

3.2. 随机微分方程

本周没有推进。

4. 下周计划

论文阅读

1. 生成模型
 - DDPM 完结
 - 薛定谔桥（精读）
 - DDIM（泛读）
2. 几何深度学习
 - General $E(2)$ - Equivariant Steerable CNNs

项目进度

1. 使用神经网络学习生命游戏的演化动力学
 - 尝试对进一步简化后的模型权重进行解释, 并用模型作为系统的演化模拟器, 统计验证所学到的规则是否正确
 - 阅读等变 CNN 的代码
 - 尝试开始做单轨道训练样本的生成、训练和演化动力学统计

理论学习

1. 随机过程课程
 - 学习完毕 Markov 过程
2. 随机微分方程
 - 完成第四章 随机积分
 - 第五章 随机微分方程 开头

参考文献

- [1] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, “Score-Based Generative Modeling through Stochastic Differential Equations,” *CoRR*, 2020, [Online]. Available: <https://arxiv.org/abs/2011.13456>
- [2] Y. Song and S. Ermon, “Generative Modeling by Estimating Gradients of the Data Distribution,” in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, Eds., 2019, pp. 11895–11907. [Online]. Available: <https://proceedings.neurips.cc/paper/2019/hash/3001ef257407d5a371a96dcd947c7d93-Abstract.html>
- [3] J. Ho, A. Jain, and P. Abbeel, “Denoising Diffusion Probabilistic Models,” *CoRR*, 2020, [Online]. Available: <https://arxiv.org/abs/2006.11239>

5. 附录

5.1. 卷积权重可视化代码

```
import numpy as np
from typing import List
from matplotlib.axes import Axes
from matplotlib import cm
import re

abbr_d = {
    'small_2_layer_seq_p4cnn': "P4CNN-small",
    'tiny_2_layer_seq_p4cnn': "P4CNN-tiny",
    'small_2_layer_seq_cnn' : "CNN-small",
    'tiny_2_layer_seq_cnn' : "CNN-tiny",
    'multiscale_0' : "MCNN",
    'multiscale_p4': "P4MCNN",
}

for idx, r in enumerate(["B3/S23", "B2/S", "B13/S012V"]):
    print("\n\n" + r)

    for s in os.listdir("."):
        p = re.compile(
            r"(?P<date>\d{4}(-\d\d){2})_(\d\d-){2}\d\d_(?P<model>.*?)__"
            r"(?P<n1>\d+)-(?P<n2>\d+)-(?P<rule>B\d*_S\d*V?)"
        )
        r_s = r.replace("/", "_")
        # print(s)
        if len(p.findall(s)) > 0:
            d = p.match(s).groupdict()
            print(d["rule"])
            # print(d["model"])
            if d["date"] in ["2025-10-16", "2025-10-17", "2025-10-18"] \
                and d["rule"] == r_s and d["model"] == model_name:

                model.load_state_dict(torch.load(os.path.join(s,
                                                                "best_simple_life_SimpleCNNSmall_0.1.0.pth")))
                state_dict = model.state_dict()
                rule = d["rule"]
                for l_idx in range(1, 4):
                    conv_layer = f"conv{l_idx}"
                    kernel = state_dict[f"{conv_layer}.weight"]
                    bias = state_dict[f"{conv_layer}.bias"]
                    col_num = 2 if kernel.shape[0] <= 4 else 4

                    fig, axs = plt.subplots(kernel.shape[0]//col_num,
                                            col_num,
                                            dpi=300,
                                            figsize=(
                                                max(col_num*3+4, 12),
                                                max(4, kernel.shape[0]//col_num*3)
                                            )
                    )

                    axs: np.ndarray|List[Axes] = axs.flatten()
                    vmin, vmax = -3, 3
                    cmap = 'RdBu'
```

```

im = None
for idx, (m, b) in enumerate(zip(kernel, bias)):
    r_arr = rearrange(m,
                       "(d c) w h -> (d w) (c h)",
                       d=(2 if m.shape[0] > 2 else 1)
                      )
    axs[idx].imshow(r_arr, cmap=cmap, vmin=vmin, vmax=vmax)
    axs[idx].set_title(f"Filter {idx}\n$b={b:.2f}$, "
                       f"$\max_i |x_i|={r_arr.abs().max():.2f}$")
    axs[idx].axis('off')
    for i in range(r_arr.shape[1]//5):
        axs[idx].add_patch(plt.Rectangle((5*i-0.5, -0.5),
                                          5,
                                          r_arr.shape[0],
                                          linewidth=1,
                                          edgecolor='black',
                                          facecolor='none'))
    for j in range(r_arr.shape[0]//5):
        axs[idx].add_patch(plt.Rectangle((-0.5, 5*j-0.5),
                                          r_arr.shape[1],
                                          5,
                                          linewidth=1,
                                          edgecolor='black',
                                          facecolor='none'))
    for txt, map_minmax in zip(['max', 'min'],
                               [np.argmax, np.argmin]):
        row, col = np.unravel_index(map_minmax(r_arr),
                                     r_arr.shape)
        # row, col = idx // r_arr.shape[0], idx % r_arr.shape[0]
        axs[idx].text(col, row,
                      f'{r_arr[row, col].item():.1f}',
                      ha='center',
                      va='center',
                      color="yellow" if abs(r_arr[row, col].item()) > vmax*0.8
                      else "black",
                      fontsize=8)
cbar_ax = fig.add_axes([0.92, 0.15, 0.02, 0.7]) # [left, bottom, width,
height]
fig.colorbar(cm.ScalarMappable(norm=plt.Normalize(vmin=vmin, vmax=vmax),
cmap=cmap),
            cax=cbar_ax,
            label='Kernel Weight Value')
plt.tight_layout(rect=[0, 0, 0.9, 1]) # 给 colorbar 留空间
plt.close()

```

5.2. VE-SDE、VP-SDE 和 sub-VP SDE 方差推导

可以这样直观理解：假设 x 都是标量随机过程，对 SMLD 的加噪过程，有

$$\begin{aligned}
 \text{Var}[x(t)] &= \text{Var}[x(t) - x(0)] \\
 &= \text{Var} \left[\int_0^t \sqrt{2\sigma\sigma'} dw \right] = \int_0^t 2\sigma\sigma' d\tau \quad \text{Itô 等式} \\
 &= \sigma^2(t) - \sigma^2(0) \quad \text{换元积分法}
 \end{aligned} \tag{9}$$

对 DDPM 的加噪过程，有一阶矩：

$$\begin{aligned}
 d\mathbb{E}[x(t)] &= \mathbb{E}[dx(t)] = -\frac{1}{2}\beta(t)\mathbb{E}[x(t)]dt \\
 \frac{d\mathbb{E}[x(t)]}{\mathbb{E}[x(t)]} &= -\frac{1}{2}\beta(t)dt && \text{分离变量} \\
 \log(|\mathbb{E}[x(t)]|) &= -\frac{1}{2}\int_0^t \beta(t)dt + C \implies |\mathbb{E}[x(t)]| = C'e^{-\frac{1}{2}\int_0^t \beta(t)dt} && \text{积分} \\
 \mathbb{E}[x(t)] &= e^{-\frac{1}{2}\int_0^t \beta(t)dt}\mathbb{E}[x(0)] && \text{假设}\mathbb{E}[x(0)] \geq 0
 \end{aligned} \tag{10}$$

二阶矩:

$$\begin{aligned}
 d\mathbb{E}[x^2(t)] &= \mathbb{E}[dx^2(t)] = \mathbb{E}[2x(t)dx(t) + \beta(t)dt] && \text{Itô 乘积法则} \\
 &= \mathbb{E}[x(t)(2\sqrt{\beta(t)}dw - \beta(t)x(t)dt) + \beta(t)dt] \\
 &= \mathbb{E}[2x(t)\sqrt{\beta(t)}dw] - \beta(t)\mathbb{E}[x^2(t)]dt + \beta(t)dt \\
 \mathbb{E}\left[\int_0^t dx^2(t)\right] &= \underbrace{\mathbb{E}\left[2\int_0^t x(t)\sqrt{\beta(t)}dw\right]}_0 - \int_0^t \beta(\tau)\mathbb{E}[x^2(\tau)]d\tau + \int_0^t \beta(\tau)d\tau && \text{期望的线性性} \\
 d\mathbb{E}[x^2(t)] &= -\beta(t)\mathbb{E}[x^2(t)]dt + \beta(t)dt && (11) \\
 \frac{d\mathbb{E}[x^2(t)]}{1 - \mathbb{E}[x^2(t)]} &= \beta(t)dt && \text{换回随机微分} \\
 -\log(|1 - \mathbb{E}[x^2(t)]|) &= \int_0^t \beta(t)dt + C \implies |1 - \mathbb{E}[x^2(t)]| = C'e^{-\int_0^t \beta(t)dt} && \text{积分} \\
 \mathbb{E}[x^2(t)] &= 1 - e^{-\int_0^t \beta(t)dt}(1 - \mathbb{E}[x^2(0)]) && \text{假设}1 - \mathbb{E}[x^2(t)] \geq 0
 \end{aligned}$$

因此方差为

$$\begin{aligned}
 \text{Var}[x(t)] &= \mathbb{E}[x^2(t)] - (\mathbb{E}[x(t)])^2 \\
 &= 1 - e^{-\int_0^t \beta(t)dt}(1 - \mathbb{E}[x^2(0)]) - e^{-\int_0^t \beta(t)dt}(\mathbb{E}[x(0)])^2 \\
 &= 1 + e^{-\int_0^t \beta(t)dt}(\text{Var}[x(0)] - 1)
 \end{aligned} \tag{12}$$

因此当 $\text{Var}[x(0)] = 1$ 时, 有 $x(t)$ 的方差恒为 1。

对于 sub-VP SDE, 套用上面的结果, 并记 $B(t) = \int_0^t \beta(s)ds$, 容易得到

$$\begin{aligned}
\mathbb{E}[x(t)] &= e^{-\frac{1}{2} \int_0^t \beta(s) ds} \mathbb{E}[x(0)] \\
\mathbb{E}[x^2(t)] &= e^{-\int_0^t \beta(s) ds} \left[\int_0^t \bar{\beta}(\tau) e^{\int_0^\tau \beta(s) ds} d\tau + \mathbb{E}[x^2(0)] \right] \\
&= e^{-\int_0^t \beta(s) ds} \left[\int_0^t \beta(\tau) \left(1 - e^{-2 \int_0^\tau \beta(s) ds} \right) e^{\int_0^\tau \beta(s) ds} d\tau + \mathbb{E}[x^2(0)] \right] \\
&= e^{-B(t)} \mathbb{E}[x^2(0)] + e^{-B(t)} \left[\int_0^t B'(\tau) (1 - e^{-2B(\tau)}) e^{B(\tau)} d\tau \right] \tag{13} \\
&= 1 + e^{-2B(t)} + e^{-B(t)} (\mathbb{E}[x^2(0)] - 2) \\
\text{Var}[x(t)] &= \mathbb{E}[x^2(t)] - (\mathbb{E}[x(t)])^2 \\
&= 1 + e^{-2B(t)} + e^{-B(t)} (\mathbb{E}[x^2(0)] - 2) - e^{-B(t)} (\mathbb{E}[x(0)])^2 \\
&= 1 + e^{-B(t)} (\text{Var}[x(0)] - 2) + e^{-2B(t)} \\
&= 1 + e^{-\int_0^t \beta(s) ds} (\text{Var}[x(0)] - 2) + e^{-2 \int_0^t \beta(s) ds}
\end{aligned}$$

一些简单的推导，可以得出 sub-VP SDE 的方差小于等于 VP-SDE 的方差。