2025 年 11 月 17 日至 11 月 23 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

目前的实现尚未考虑改变等变约束，实现的是其他所有模块的功能。其大致是算法结构如下

```
# read and process the command line arguments
# initialize dataset and dataloader with only one visible trajectory.
# initialize RuleSimulator
# initialize predictor model.

# while prediction loss and simulation loss is not converged
for round_id in range(1, 21):
    # train predictor on visible trajectories.
    # update rules in RuleSimulator from predictor.
    #randomly select one of the invisible trajectories to simulate.
    #evaluate predictor on one of the visible trajectories.
    # calculate simulation acc.

    if sim_acc > 0.99 and train_loss < 0.01 and evaluate_acc > 0.95:
        # break
    else:
        # randomly add one trajectory
# save predictor model.
# save rules.
```

2. 文献阅读

2.1. Stable Neural SDE in Analyzing Irregular Time Series Data [1]

ICLR 2024 | YongKyung Oh et al.

本论文讨论了提出了一些保证解的唯一性、随机稳定性和数值稳定性的 Neural SDE 模式。

2.1.1. Neural ODE 和 Neural SDE 简介

Neural ODE 将神经网络作为导数 $\frac{\mathrm{d}f}{\mathrm{d}x}$ 的预测器。令 $h(x;\theta_h)$ 将数据点投影至隐空间（对应地还得有一个 decoder），得到隐空间中的表示 z，隐空间中的 ODE $\mathrm{d}z = f\big(t, z(t); \theta_f\big)\mathrm{d}t$ 的解就可以写为

$$z(t) = z(0) + \int_0^t f\big(\tau, z(\tau); \theta_f\big)\mathrm{d}\tau, \qquad z(0) = h(x;\theta_h) \tag{1}$$

其中 f 和 h 都是神经网络。Neural CDE (neural controlled differential equation) 添加了一个控制信号 X，它常常是数据点的插值曲线。Neural CDE 将 Neural ODE 中的积分改为 Riemann-Stieltjes 积分：

$$z(t) = z(0) + \int_0^t f\big(\tau, z(\tau); \theta_f\big)\mathrm{d}X(\tau), \qquad z(0) = h(x;\theta_h) \tag{2}$$

Neural SDE 的思想也很类似，它将偏移项 f 和扩散项 g 都由神经网络代替，其求解形式为

$$z(t) = z(0) + \int_0^t f\big(\tau, z(\tau); \theta_f\big)\mathrm{d}\tau + \int_0^t g\big(\tau, z(\tau); \theta_s\big)\mathrm{d}W(\tau), \qquad z(0) = h(x;\theta_h) \tag{3}$$

其中 $W(t)$ 是和 z 维数相同的 Brown 运动。我们也可以仿照 Neural CDE 的形式，为 Neural SDE 引入一条控制轨道。令

$$\bar{z}(t) = \zeta\big(t, z(t), X(t); \theta_\zeta\big) \tag{4}$$

然后将 $z(t)$ 替换为 $\bar{z}(t)$。

有了上述的积分形式，**Neural ODE 和 SDE 的求解可以交由数值算法完成。神经网络参与进来的点是估计偏移项和扩散项（若有），训练好的神经网络可以参与数值计算过程，当做原本的偏移项函数和扩散项函数来用。**引入随机项使得 Neural SDE 面临三大难题，分别是解的存在**唯一性、随机稳定性和数值稳定性**。如果某形式的 SDE 缺乏唯一强解，同一初值可能经过计算进入完全不同的轨道，使得训练变得困难。引入的随机项会使得得到的轨道不平稳，可能导致训练过程中的梯度爆炸。另外，随机项的引入会使得数值计算中的稳定性更加重要，需要设置更加数值稳定的 SDE 形式。

2.1.2. 良好性质的 Neural SDE 结构

作者提出了三个 Neural SDE 结构，分别是 **Langevin 型 SDE**、**线性噪声 SDE** 和 **几何 SDE**。其形式分别如下

$$\begin{aligned}
&\mathrm{d}z(t) = \gamma\big(z(t);\theta_\gamma\big)\mathrm{d}t + \sigma(t;\theta_\sigma)\mathrm{d}W(t) && \text{Langevin 型 SDE (LSDE)}\\
&\mathrm{d}z(t) = \gamma\big(t, z(t);\theta_\gamma\big)\mathrm{d}t + \sigma(t;\theta_\sigma)z(t)\mathrm{d}W(t) && \text{线性噪声 SDE (LNSDE)}\\
&\frac{\mathrm{d}z(t)}{z(t)} = \gamma\big(t, z(t);\theta_\gamma\big)\mathrm{d}t + \sigma(t;\theta_\sigma)\mathrm{d}W(t) && \text{几何 SDE (GSDE)}
\end{aligned} \tag{5}$$

这三种 Neural SDE 结构理论上可以解决先前提到的三个问题，并在实验中表现良好。

3. 学习进度

3.1. 随机过程

本周继续系统学习 Markov 链，了解了常返性的一些性质。特别地，在有限状态的时齐 Markov 链中，相互联通的节点常返性相同。

3.2. 随机微分方程

本周开始学习 SDE 解的存在性和唯一性。

3.3. 量子力学

了解了量子力学的一些基本概念，例如波函数、薛定谔方程、无限深势阱、算符。

4. 问题记录

4.1. SDE 数值解的问题

我尝试使用 Python 求解下面的 SDE:

5. 下周计划

论文阅读

1. 生成模型
   - 薛定谔桥
   - DDIM

项目进度

1. 使用神经网络学习生命游戏的演化动力学
   - 调试完成代码，并考虑等变约束
2. 耦合约瑟夫森结
   - 将 MATLAB 模拟代码全部迁移至 Python
   - 考虑简单的 Neural SDE 方法解带参 OU 过程的参数

理论学习

1. 随机过程课程
   - 复习 Poisson 过程和 Markov 过程
2. 随机微分方程
   - 第五章

参考文献

[1] Y. Oh, D.-Y. Lim, and S. Kim, “Stable Neural Stochastic Differential Equations in Analyzing Irregular Time Series Data.” [Online]. Available: https://arxiv.org/abs/2402.14989