2025 年 11 月 24 日至 11 月 30 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. 训练数据可视化和裁剪

本周实现了可视化训练得到轨道的图形化界面程序。发现对于部分规则（例如 B3678/S34678）容易演化至平凡的不动点，因此可以控制在生成轨道的过程中，如果发现系统落入一阶不动点，那么就终止模拟。这样做可以提高得到数据集的质量，然而这样操作会使得数据集读取过程变得稍复杂。

1.1.2. 优化规则提取器

1.1.2.1. 统计存活邻居细胞数量的方法（黑盒方法）

一个暂行的方法是做下面和经典的生命游戏完全相同的假定：一是下一状态只依赖于这个细胞本身和它周边的八个细胞，二是细胞周边的八个邻居细胞对细胞下一状态的权重相同。这样我们可以用神经网络作为演化器，对若干随机输入批量演化下一状态（随机的输入可以是形状为 `[N, 1, w, h]` 的张量），然后统计状态转移 $(\boldsymbol{x}_t, \mathcal{N}(\boldsymbol{x}_t)) \to \boldsymbol{x}_{t+1}$。这会得到对应的条形图：

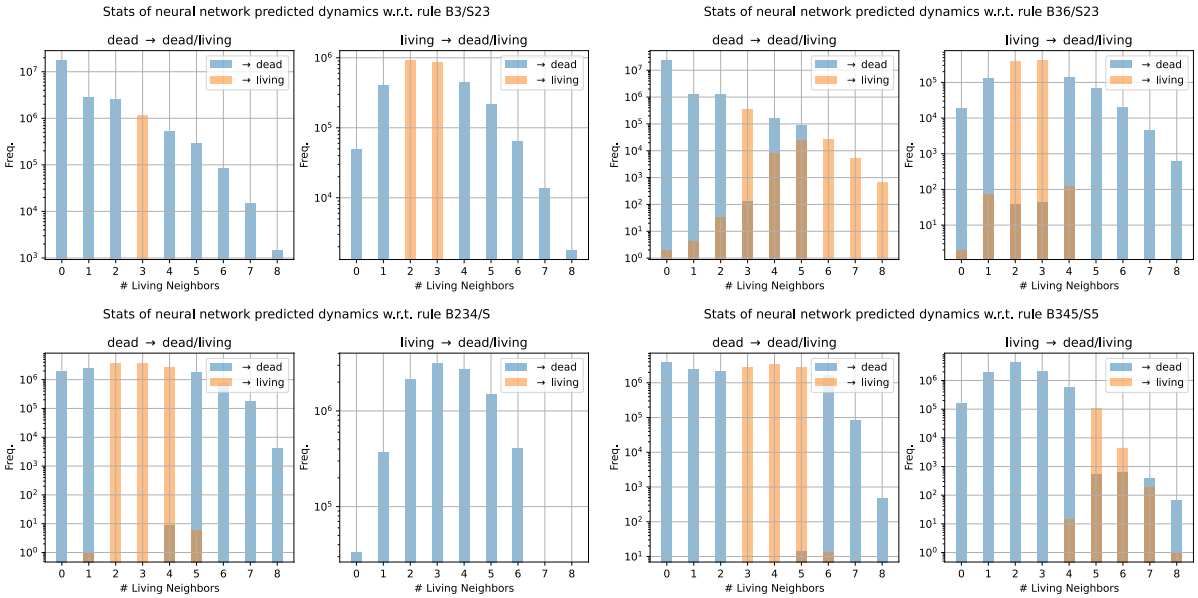

图 1　部分 Moore 邻域规则在 3 层神经网络上训练后的统计结果

此外可以使用例如 FixMatch 这样的半监督的方法，对于高置信度的转移规则，将其确定为神经网络所学到的系统演化规则。可以看到对于存活和诞生对邻居细胞数量的要求，连续的情形（例如 B3/S23，指的是 B.../S... 中的整数列是否是连续的，如 123 是连续的，而 124 不连续）相比不连续的情形（例如 B36/S23）更容易学到高确信且正确的结果：从上图中左上角、左下角的结果可以直接推出系统的演化规则。

但对于规则不连续，或者规则中包含较大的数字（例如 B3678/S35678 这样的规则）时，模型即使在训练时的正确率超过 99%，在较大的邻居存活数的预测情况相对不佳。从上图可以看到，存活邻居为 5 个以上的计数大致是总数的百分之一以下。

1.1.2.2. 遍历输入特征的映射方法（黑盒方法）

若不添加除平移不变性以外的任何对称性约束，并想通过穷尽所有的输入邻居以得到网络提取的结果，需要考虑 $2^{|\mathcal{N}(\boldsymbol{x})|}$ 左右的情况，而这将在邻域扩大时变得不可能。若在训练好的模型权重能发现有关邻域的特征，例如 Moore 邻域的每个邻居细胞对中心细胞的贡献相似，或是存在类似的大前提，则上述考虑的邻域数量

1.1.2.3. 从神经网络权重出发的方法（白盒方法）

从神经网络权重出发的主要思路为将一个复杂规则，例如 B3678/S34678 提取得到一系列的子规则，类似于空间的基。假设我们得到了可以构建所有规则的一个“基规则集”，并在该集合中的每条规则上训练一个和耦合规则一样结构的神经网络，对比训练完成后网络的权重，我们期望参与构成耦合规则的基规则网络的权重能在耦合规则网络的权重中有所体现。检查的方法目前想到的有两个。一是定性的，直接对权重进行可视化，然后比较可视化所见权重：基规则中权重模式是否出现在耦合规则网络的权重中；二是定量的，构造参数空间中的某种内积，然后做内积，得到耦合网络权重在各基规则网络权重方向的“分量”。也许后者可以通过类似度量学习一样，学习一个内积网络出来。

目前该规则中拆分或得到基规则是个难题。如果直接按照 naive 的想法，切成 B1/S, …, B8/S, B/S1, …, B/S8，模拟生成时对于大部分的规则，将极其容易跌至平凡不动点，可用于训练的数据极少。

2. 文献阅读

2.1. Back to Basics: Let Denoising Generative Models Denoise

本文将直接预测 $\boldsymbol{x}$，而不是如 DDPM 那样直接预测 $\boldsymbol{\epsilon}$ 或是速度向量 $\boldsymbol{v}$ 那样间接生成 $\boldsymbol{x}$，并指出了这样做的优越性。其理论依据为机器学习中的低维流形假设，因此作者认为直接预测 $\boldsymbol{x}$ 和预测速度或是噪声有本质的不同。进一步地，作者提出建模这个低维流形的难度低于建模全空间中的噪声或是速度场。

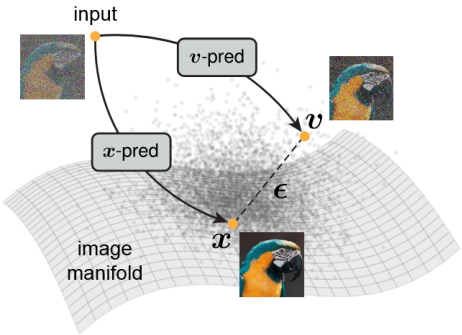

图 2 数据的流形假设决定预测真实照片 $\boldsymbol{x}$ 和预测噪声 $\boldsymbol{\epsilon}$ 或预测速度 $\boldsymbol{v}$ 完全不同

为显示直接预测 $\boldsymbol{x}$ 的优越性，作者考虑一个二维中的一维流形上的数据，然后使用一个随机的投影矩阵 P 将其投影至高维空间 $\mathbb{R}^D$，然后使用预测 $\boldsymbol{x}$、预测噪声或预测速度的模型生成来自该流形上分布的高维数据，再通过投影变换投影到二维平面，结果如下图所示。可以看见只有直接预测 $\boldsymbol{x}$ 的模型在升高维度后依然保持较好的性能，而剩余两种当维度超过 16 后结果很差。

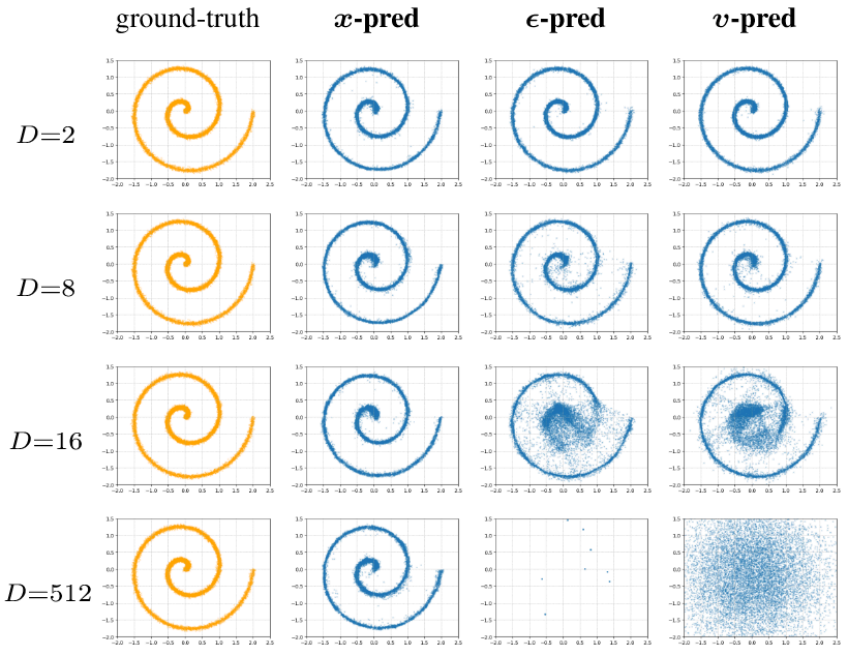

图 3 不同预测目标模型在学习流形分布小实验下的结果

作者将提出的模型称为 **JiT (Just image Transformer)**，其架构就是一般的 ViT (Vision Transformer)，如下图左侧所示。其原理为将图片划分成若干网格，然后取每格网格中内容经过一个 Projection Embedding 之后加上 Position Emedding 作为 Transformer 模块的输入，然后堆叠 Transformer 模块，最后取输出再投影回原来的形状，最后拼接成生成的图像。JiT 使用的损失函数是对应于速度的损失函数：

$$\mathcal{L} = \mathbb{E}_{t,\boldsymbol{x},\boldsymbol{\varepsilon}} \|\boldsymbol{v}_\theta(\boldsymbol{z}_t, t) - \boldsymbol{v}\|^2$$
$$\boldsymbol{v}_\theta(\boldsymbol{z}_t, t) = \frac{\mathrm{Net}_\theta(\boldsymbol{z}_t, t) - \boldsymbol{z}_t}{1 - t} \tag{1}$$

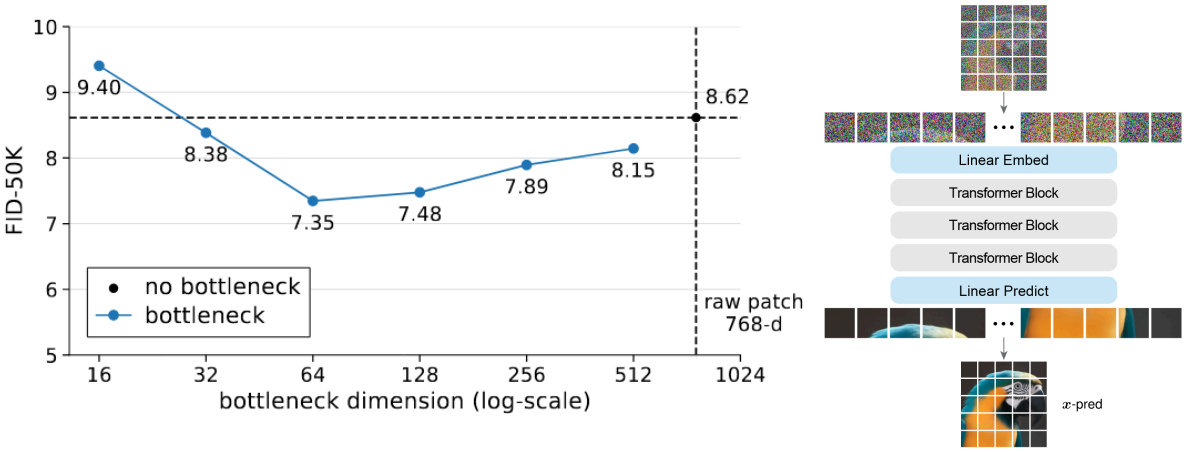

图 4 （左）模型性能与 BottleNeck 宽度的关系（右）ViT（或 JiT）的主干结构

作者用先前的 Baseline 和 JiT 模型在 ImageNet 上进行测试，发现 JiT 拥有最好的性能，并得出了下面的若干结论：

1. 直接预测 $\boldsymbol{x}$ 很关键
2. 损失函数中的权重无关紧要
3. 调整流匹配中的 t 的采样分布不关键
4. 增加网络宽度不必要
5. 在网络中添加一定宽度的 BottleNeck 结构可以生成分数更高

这篇文章的主要启示是费尽心思为生成设计的 **SDE** 是否落入了思维陷阱，这值得仔细思考。另一方面，这似乎又宣告“漂亮的数学模型”在生成这一领域的一次失败？

3. 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
   - 考虑另外两种方法的实现
   - 更新在线 Overleaf 文档
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
   - 第五章完成