Neural Jump Stochastic Differential Equations

Junteng Jia and Austin R. Benson · Cornell University

Motivation & Problem Statement

Many real-world systems evolve continuously over time but are interrupted by stochastic events. For example, a social network user might have some evolving interest in a product that is abruptly changed by seeing an ad. How can we simultaneously learn continuous and discrete dynamics?

Given:
- $\mathcal{H}_t = \{(\tau_i, k_i)\}_{k=1}^t$ — events up to time $t$; $\tau_j$ is a timestamp and $k_j$ is an (optional) discrete or continuous label

Goal:
- Learn the latent dynamics that generated $\mathcal{H}_t$
- Predict the likelihood or label of future events

Background on Point Process Models

We model event sequences with point processes, where event generation is described by a conditional intensity:

$$P\{\text{event in } [t, t+dt) \mid \mathcal{H}_t\} = \lambda(t) \cdot dt$$

Intensity dynamics depend on $\mathcal{H}_t$ and can be written as a jump SDE. If $N(t)$ counts the number of events before $t$:

$$d\lambda(t) = \beta \cdot [\lambda(t) - \lambda_0] \cdot dt + \alpha \cdot dN(t)$$

Limitation: the functional form of $\lambda(t)$ dynamics for must be provided. Some widely-used function forms shown above.

Model and Learning

We follow the ideas of Neural ODEs\(^1\) and parameterize the jump SDE model with neural nets and a latent $z(t)$. This gives our neural jump SDE model (NJSDE):

$$dz(t) = f(z(t), \theta) \cdot dt + w(z(t), \theta) \cdot dN(t)$$

$$\lambda(t) = \lambda(z(t), \theta)$$

We can use learned latent continuous dynamics $z(t)$ for simulation and prediction.

Training with the adjoint method\(^2\) (here just to compute the gradient $\delta \mathcal{L} / \delta a(t_0)$ = $a(t_0)$)

1. For desired loss or likelihood $\mathcal{L}$, set $a(t_N) = \delta \mathcal{L} / \delta z(t_N)$
2. Integrate $a(t) = -a(t) \frac{\partial f(z(t), \theta)}{\partial z(t)}$ backwards until event at $\tau$
3. Update $a(\tau) = a(\tau^+) + a(\tau^-) \frac{\partial f(z(\tau), \theta)}{\partial z(\tau)}$
4. Go to step 2

By augmenting $z(t)$ to include $\theta$, this method can be used to learn all of the latent dynamics. (See paper for details.)

Learning true conditional intensities

- **Input**: event sequences from classical point processes
- **Output**: accurately learned conditional intensities $\lambda(t)$, as measured by mean absolute percentage error (MAPE)

<table>
<thead>
<tr>
<th></th>
<th>MAPE</th>
<th>Hawkes (E)</th>
<th>Hawkes (PL)</th>
<th>Self-Correcting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hawkes (E)</td>
<td>3.5</td>
<td>155.4</td>
<td>29.1</td>
<td></td>
</tr>
<tr>
<td>Hawkes (PL)</td>
<td>128.5</td>
<td>9.8</td>
<td>29.1</td>
<td></td>
</tr>
<tr>
<td>Self-Correcting</td>
<td>101.0</td>
<td>87.1</td>
<td>29.1</td>
<td></td>
</tr>
<tr>
<td>RNN</td>
<td>22.0</td>
<td>20.1</td>
<td>24.3</td>
<td></td>
</tr>
<tr>
<td>NJSDE</td>
<td>5.9</td>
<td>17.1</td>
<td>9.3</td>
<td></td>
</tr>
</tbody>
</table>

The NJSDE can learn complex delaying effect of power-law Hawkes process with interacting latent dimensions (panel D).

Predicting continuous outcomes (synthetic)

Event labels are sampled from a distribution $k \sim p(k | z(t), \theta)$. Our model can predict labels with mean absolute error 0.35, an order of magnitude lower than predicting the mean (3.65).

Predicting discrete outcomes (Web / medical data)

Each event sequence is the awards history of a Stack Overflow user or the clinical visit history of a patient. The goal is to predict the award type or visiting reason for each event.