Report for Point Process Assignment 2

Zhaoyu Li

June 22, 2020

1 Introduction

This project is derived from Predictive Maintenance Problem. It involves equipment risk prediction to allow for proactive scheduling of corrective maintenance. Such an early identification of potential concerns helps deploy limited resources more efficiently and cost effectively, reduce operations costs and maximize equipment uptime. In this assignment, I use ADMM-MM algorithm to build a multi-dimensional hawkes model based on a real-world dataset of more than 1,000 automated teller machines (ATMs) from a global bank headquartered in North America, and predict when a machine encounters an error and what the error is.

2 Prerequisites

I implement the assignment using Python API. To run my code, **numpy**, **csv** and **easydict** should be installed. All the parameters are stored in **utils/config.py**. You can use your own parameters by changing **experiment.yaml** if you like. Run **python assignment2.py – cfg experiment.yaml** to use ADM4 algorithm to fit on the training dataset and predict on test dataset. A and μ are stored in **output/A.pkl** and **output/mu.pkl**. The results are showed in Section 4.

3 Encountered Problems

When trying my first version of implementation, I encounter some problems as below:

- In the first loop, Z_1 and Z_2 are not initialized but they are required when updating A and μ.
- I am confused with the concepts of $G(t)$ and $g(t)$.
- The minimum time of dataset is very large but the difference between the maximum time and minimum time is small.
- There are some very long event sequences in the training dataset which makes training procedure really slow.

After asking to TA, I change the order of calculation in ADM4 algorithm to initialize Z_1 and Z_2 and make clear the concept of $G(t)$ and $g(t)$ as well. To address the third issue, I normalize the time of all the data by subtracting all the time a large number which is close to the minimum time. I also discard the event sequences in the training dataset whose length are longer than 1e5 to speed up the training. (Note that there is **no** modifying on test dataset!)

However, when I run the program without an error, it takes very long time to finish a event sequence. Since I use many loops to calculate A and μ iteratively, the running time of the training procedure is unacceptable. After discussing with classmates and consulting from the codebases online, I use matrix multiplication instead of loops to calculate p_{ii} and p_{ij}. Specifically, the summation in the denominator $\sum_{j=1}^{i-1} a^{(m)}_{u_i^c u_j^c} g(t_i^c - t_j^c) = \sum_{k=1}^{7} a^{(m)}_{u_i^c k} \sum_{j=1, u_j^c = k}^{i-1} g(t_i^c - t_j^c)$ can be calculated by matrix multiplication. Denote $\sum_{j=1, u_j^c=k}^{i-1} g(t_i^c - t_j^c)$ as $f[i][k]$, then $\sum_{j=1}^{i-1} a^{(m)}_{u_i^c u_j^c} g(t_i^c - t_j^c) =< A[u_i^c], f[i] >$, where $<, >$ indicate inner product. Since f can be calculated only once, it's efficient to calculate p_{ii} and p_{ij}. Also instead of calculate B and C by calculating its value on each position, I use a matrix to record each value and traverse all the event sequences only once. These version of implementation is much faster than the first version that it takes less than two hours for training.

4 Results and Analysis

In this assignment, I use two metrics to measure the performance of my model. The first one is "Event Type Prediction", which calculates precision, recall, macro-F1 score over 7 event types and then average over all types. The second one is "Event Time Prediction", which is the mean absolute error (MAE) which measures the absolute difference between the predicted time point and the actual one. The MAE of event time prediction is **0.499** and the results of event type prediction are showed as the Table 1 below:

Table 1: Event Type Prediction of the model

	PRT	CNG	IDC	COMM	LMTP	MISC	TIKT	avg
Precision	0.476	0.906	0.608	0.640	0.803	0.932	0.015	0.626
Recall	0.543	0.916	0.590	0.581	0.805	0.932	0.007	0.625
macro-F1 Score	0.507	0.911	0.599	0.609	0.804	0.932	0.009	0.624

Almost all of the parameters I use is the same as the model in [Zhou et al., 2013], e.g $\lambda_1 = 0.02$, λ_2 = 0.6. However, the weight decay w is not listed in the paper. When I set $w = 0.01$ as in assignment 1, the performance drop a lot that almost all of the metric scores are below 0.1. It seems like that the weight decay is very important to the model, especially to fit on a large, diverse, real-world dataset. If the weight decay is too small, the influence of former events exert a tremendous influence on the next event, which makes a lot of noise to the prediction. So I use a small grid search to find a better w and finally I set $w = 1$ which leads to the results now.

As for the result, the prediction of error type TIKT is very bad since that there is few training samples in the training dataset and the appearance of TIKT is irregular that it's hard for the model to predict. On the opposite, the prediction of CNG, LMTP and MISC is quite well. It is because they often continuously appear in an event sequence and the model may learn the pattern of their appearances very well.

References

Ke Zhou, Hongyuan Zha, and Le Song. Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes. In *Artificial Intelligence and Statistics*, pages 641–649, 2013.