Foundations of DS: Cohortney project

Taras Khakhulin, Mikhail Pautov

Second project report, April 2021

1 Problem statement

The problem of clustering of event sequences may be formulated as follows. Suppose we have a set of event sequences $S = \{s_n\}_{n=1}^{N}$, where $s_n = \{(t_i, c_i)\}_{i=1}^{M_n}$ contains a series of events $c_i \in C = \{1, ..., C\}$ and their time stamps $t_i \in [0, T_n]$. The goal is to cluster set S by splitting it onto an optimal (in some sense) number of non-intersecting subsets in which there are similar (in some sense) event sequences. It is supposed that the characterization of point processes is done via intensity function $\lambda_c(t) = \mathbb{E}\frac{d}{dt}N_c(t)|\mathcal{H}_t^C$, where $N_c(t)$ is the number of type-c events occurred before moment t and $\mathcal{H}_t^C = \{(t_i, c_i)|t_i < t, c_i \in C\}$.

The aim of this project is to refactor two existing repositories of sequence clustering (6 methods of clustering to be considered at the best) and combine them into a single library with a standard API. The framework itself based on the Pytorch Lightning.

2 Description of a baseline solutions

There are 6 methods of clustering of event sequences to be considered in the project:

- Pure Cohortney (clustering based over partitions).
- Cohortney (LSTM and EM over partitions).
- Dirichlet Mixture Model of Hawkes Processes.
- Deep Clustering over Cohortney.
- Convolutional Autoencoder Clustering over Cohortney.
- Optimal Transport for Clustering over Cohortney.

3 Main challenges

One of the main problem is the difference of the API for all presented methods. Every method except one does not have clear code base and the task of the refactoring into single API is not easy at first glance. The difference between two of the methods is no clear. We are going to reimplement the first method as our baseline for API and create the overall infrastructure around it.

3.1 Data

We use synthetic data and real datasets from Cohortney paper without generating new datasets for clarity.

3.2 Framework details

In our Framework we implement the 6 methods with similar api using trainer from PyTorch Lightning as the main framework for the library with Black formating. We are going to support the hyperparameters tunning with Ray and documentation using readthedocs. We want to design simple notebook with tutorial of Cohortney usage. The overall framework will be availiable using pip.

4 Distributions of the roles of participants

The initial distribution of the workload (according to the aims in a spreadsheet) is as follows:

- Taras: tests, package, ray methods.
- Misha: docs, formating, trainer, api.

5 Project structure overview

In this section, we provide the structure of the repository as well as list of external libs.

The structure of the repository. Overall, current structure of the repository looks as follows:

```
├── README.md
├── configs
│   ├── callbacks
│   │   └── default.yaml
│   ├── config.yaml
│   ├── datamodule
│   │   └── default.yaml
│   ├── experiment
│   │   └── default.yaml
│   ├── logger
│   │   └── default.yaml
│   ├── model
│   │   └── default.yaml
│   └── trainer
│       └── default.yaml
├── reports
├── requirements.txt
├── reviews
├── run.py
└── src
    ├── __init__.py
    ├── dataset
    │   ├── __init__.py
    │   └── random_seq.py
    ├── dataset_generator.py
    ├── model
    │   └── single_pp_cohortney.py
    ├── networks
    │   ├── losses.py
    │   └── lstm_pp.py
    ├── train.py
    └── utils
        ├── __init__.py
        ├── base.py
        ├── datamodule.py
        ├── metrics.py
        └── net.py
```

Libraries in usage: The following libs are used in the project so far: *pytest, scikit-learn, pandas, matplotlib, seabor, hydra-core, torch, torchvision, pytorch-lightning, torch-optimizer, test-tube.* The necessity to use them is obvious – DL frameworks for scalable and modulable coding, visualization, data preprocessing and testing.

6 Link to the GitHub repository

All the project work will be conducted in the scope of the corresponding ADASE repository (clickable or use the link https://github.com/adasegroup/cohortney).