Cohortney: the framework for clustering of event sequences

Taras Khakhulin, Mikhail Pautov

Problem statement

The problem of clustering of event sequences is formulated as follows:

- Set of event sequences *S* contains of objects *s* = *{(t, c)}* as sequences of occurrences of *c*-type events (out of C possible types, *{1,2,...,C}*) at time points *t*.

- 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 – expected grow of *c*-type event at time *t* given number of its occurrences before *t*.

Our goal is to build a **framework** for methods solving such a problem.

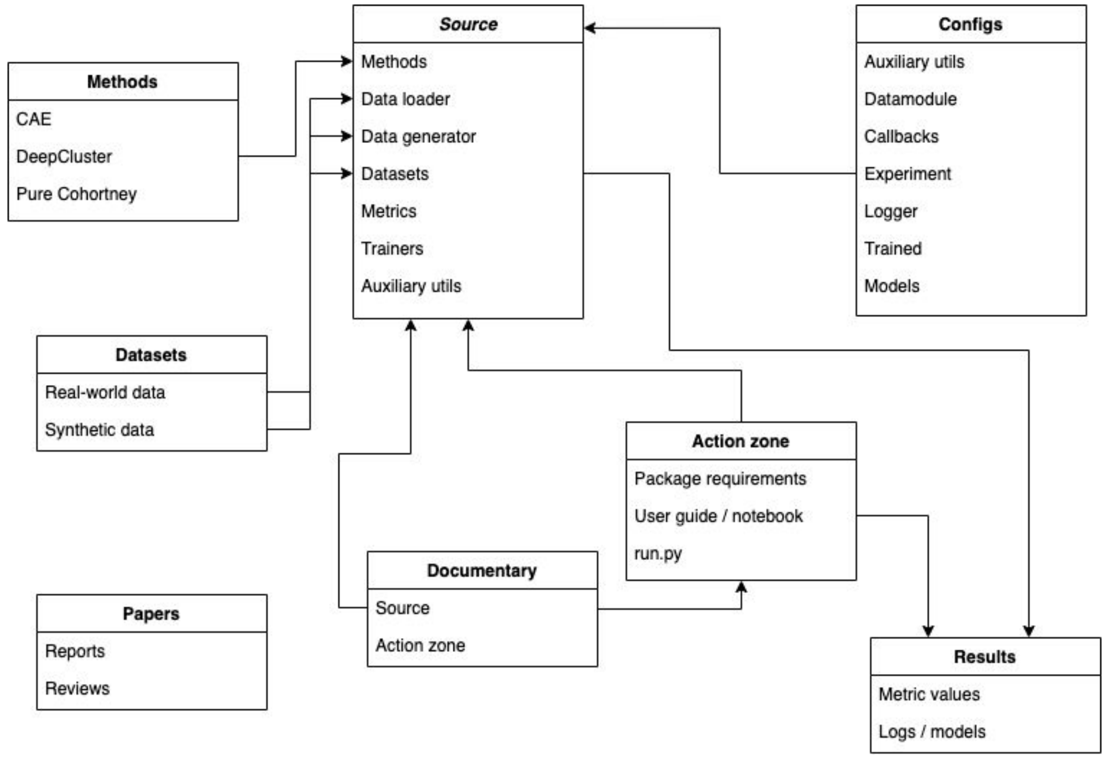
Potential application / use-cases

- The framework as a tool: testing and evaluation of different methods of event clustering

- The methods' application
    - Banking sphere – bank transactions, user actions monitoring
    - Business processes – prediction / preempt of failure regarding certain actions
    - Healthcare system – monitoring of medical manifestations / features of patients

- Clustering in a nutshell
    - Websites' usage monitoring
    - Clustering visitors of different institutions

Structure of framework



- modular structure
- all modules are configurable
- different methods are implemented
- single API structure
- built-in data loader, downloader and preprocessor
- properly documented
- transparent pipeline
- user guide inside

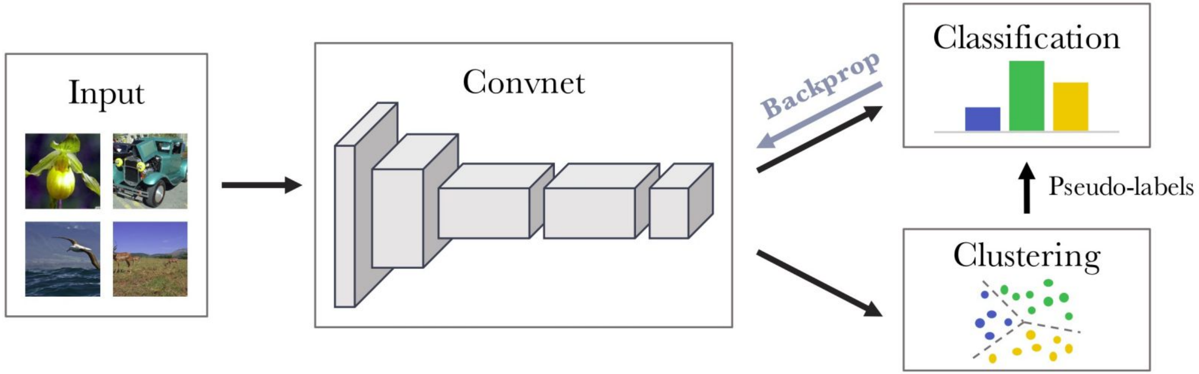
Skoltech

Methods implemented

- Pure Cohortney
  - The main idea of this method is the following: clustering based on the LSTM Model
    We have **K** clusters and **N** realizations of event sequences.
    Given these training sequences, we want to find the clusters among them and the corresponding intensity functions.
- Deep Clustering over Cohortney
- Convolutional Autoencoders over Cohortney

Methods implemented

- Pure Cohortney
- Deep Clustering over Cohortney
    - Given partitions obtained from Cohortney for a particular **N** we may proceed with clustering algorithms suitable for sequences of features of the same length.
      Therefore we use combination Cohortney with a deep learning approaches for object clustering, such method is called Deep Clustering.
- Convolutional Autoencoders over Cohortney



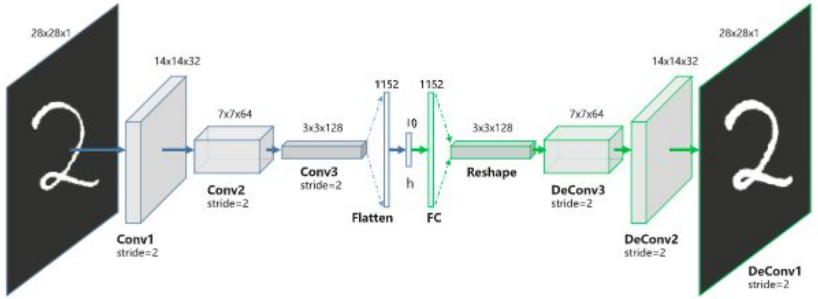
Skoltech

Methods implemented

- Pure Cohortney

- Deep Clustering over Cohortney

- Convolutional Autoencoders over Cohortney
  - The clustering of representation with convolutional autoencoders is well-known and widely used method. we apply 1D convolutions over cohortney estimated over event sequences.



Skoltech

Datasets description

Automatically Downloaded feature

- Synthetic
  - sine-like impact function, which piecewise constant
  - exponential kernel function

- Real Datasets
  - AgeDataset
    - Contains sequences of transaction records stemming from clients at financial institutions. For each client, we have a sequence of transactions. Each transaction is described with a discrete code, identifying the type of transaction, the Merchant Category Code, such as a bookstore, ATM, drugstore, and more. According to transactions' history, dataset is categorized clients into age categories by the authors. The data provides each client's age bin with four different bins in total.
  - LinkedIn
    - Contain users' history of employers. There are 82 different companies, comprising different event types, and with 2439 users in total. Based on this data, we can categorize users into different professional areas. There is information about industry sectors for each user to compare those sectors with real ones. The event sequences in these three datasets have strong but structural triggering patterns, which we can model via different Hawkes processes.

[1] Zhuzhel V. et al. COHORTNEY: Deep Clustering for Heterogeneous Event Sequences //arXiv preprint arXiv:2104.01440. – 2021.

Validation procedure

We follow the evaluation procedure described in the original paper [1]: when the labels are obtained by a certain method, we use one out of two metrics to evaluate the results:

- *Consistency,* or minimum proportion of preserved pairs over trials (for real-world data):

$$\text{Consistency} = \min_{j \in \{1,\dots,J\}} \sum_{j' \neq j} \sum_{(n,n') \in \mathcal{M}_j} \frac{1\left\{k_n^{j'} = k_{n'}^{j'}\right\}}{(J-1)\,|\mathcal{M}_j|}$$

- *Purity,* (for synthetic data) defined as $\text{Purity} = \frac{1}{N} \sum_{k=1}^{K} \max_{j \in \{1,\dots,K'\}} |\mathcal{W}_k \cap \mathcal{C}_j|$

Here $\mathcal{M}_j = \left\{(n,n') \mid k_n^j = k_{n'}^j\right\}$ is the set of sequence pairs within the same cluster in j-th trial, k_j^n is index of cluster of n-th sequence in the j-th trial, W_k is the learned index set of sequences belonging to k-th cluster, C_j is the real index set of sequences belonging to j-th cluster and N is the total number of sequences

[1] Zhuzhel V. et al. COHORTNEY: Deep Clustering for Heterogeneous Event Sequences //arXiv preprint arXiv:2104.01440. – 2021.

Results: excerpt

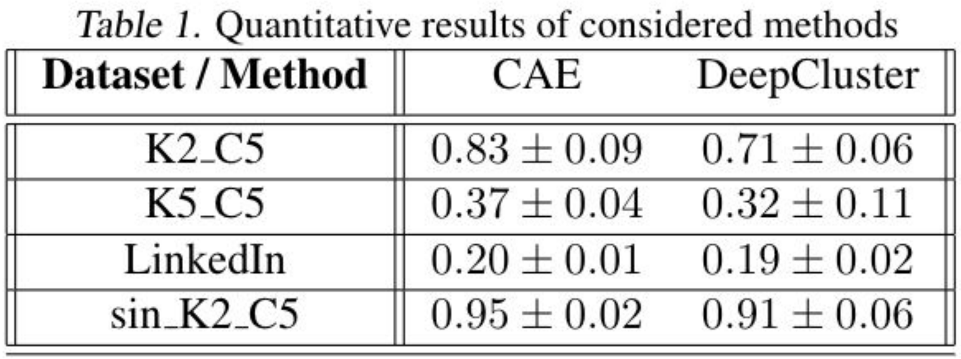


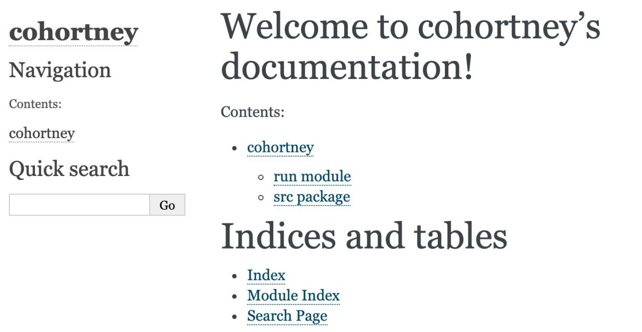
Table 1. Quantitative results of considered methods

Dataset / Method	CAE	DeepCluster
K2_C5	0.83 ± 0.09	0.71 ± 0.06
K5_C5	0.37 ± 0.04	0.32 ± 0.11
LinkedIn	0.20 ± 0.01	0.19 ± 0.02
sin_K2_C5	0.95 ± 0.02	0.91 ± 0.06

- The results are mainly illustrative
- The columns correspond to methods, the rows correspond to datasets
- The value of Purity metric is on intersection

- Clustering with use of autoencoders over Cohortney is better than with DeepCluster over Cohortney
- The synthetic data is easier to be clusterized
- Increasing the number of GT clusters affects performance of a model
- Runs of the methods are hyperparameters-dependent

Documentation



cohortney

Navigation

Contents:

cohortney

Quick search

Go

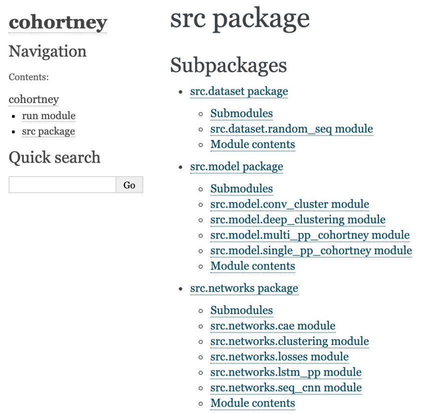
Welcome to cohortney's documentation!

Contents:

- cohortney
  - run module
  - src package

Indices and tables

- Index
- Module Index
- Search Page



cohortney

Navigation

Contents:

cohortney
- run module
- src package

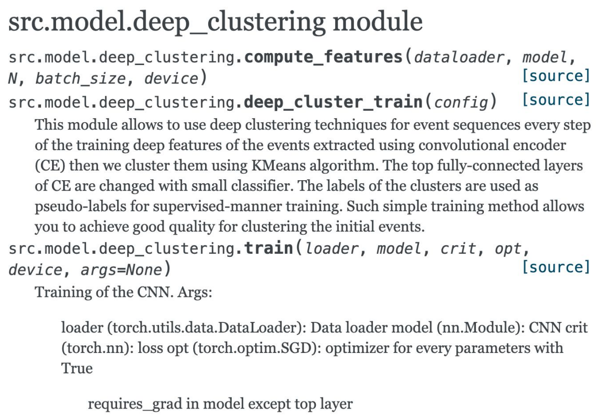
Quick search

Go

src package

Subpackages

- src.dataset package
  - Submodules
  - src.dataset.random_seq module
  - Module contents
- src.model package
  - Submodules
  - src.model.conv_cluster module
  - src.model.deep_clustering module
  - src.model.multi_pp_cohortney module
  - src.model.single_pp_cohortney module
  - Module contents
- src.networks package
  - Submodules
  - src.networks.cae module
  - src.networks.clustering module
  - src.networks.losses module
  - src.networks.lstm_pp module
  - src.networks.seq_cnn module
  - Module contents



src.model.deep_clustering module

`src.model.deep_clustering.compute_features(dataloader, model, N, batch_size, device)` [source]

`src.model.deep_clustering.deep_cluster_train(config)` [source]

This module allows to use deep clustering techniques for event sequences every step of the training deep features of the events extracted using convolutional encoder (CE) then we cluster them using KMeans algorithm. The top fully-connected layers of CE are changed with small classifier. The labels of the clusters are used as pseudo-labels for supervised-manner training. Such simple training method allows you to achieve good quality for clustering the initial events.

`src.model.deep_clustering.train(loader, model, crit, opt, device, args=None)` [source]

Training of the CNN. Args:

loader (torch.utils.data.DataLoader): Data loader model (nn.Module): CNN crit (torch.nn): loss opt (torch.optim.SGD): optimizer for every parameters with True

requires_grad in model except top layer

- The whole project is properly documented
- Documentation provides information about functionality and description of all the modules in a project
- Documentation also consists of guides and pieces of advices for framework's users

Conclusion

- proposed a framework for event sequences clustering methods which have a lot of industrial applications.
- easy reproducibility for such unique task.
- extends for new dataset without any problems
- adding new methods is now possible without difficulties, as the functionality has become more unified

Conclusion

- proposed a framework for event sequences clustering methods which have a lot of industrial applications.
- easy reproducibility for such unique task.
- extends for new dataset without any problems
- adding new methods is now possible without difficulties, as the functionality has become more unified

Possible Improvements:

Add More:

- metrics
- methods
- datasets

Conclusion

- proposed a framework for event sequences clustering methods which have a lot of industrial applications.
- easy reproducibility for such unique task.
- extends for new dataset without any problems
- adding new methods is now possible without difficulties, as the functionality has become more unified

Possible Improvements:

Add More:

- metrics
- methods
- datasets

Takeaways:

- ML is not about fit and predict
- some tasks has unique specification and difficult to understanding