Machine Learning course 2023. Final Project 11.

ClosedAI

Contrastive Learning for Event Sequences with Self-Supervision on multiple domains

Replication study

Denis Grankin Ekaterina Andreichuk Ivan Apanasevich

Irena Gureeva Mikhail Konenkov

Motivation

ClosedAI

Learning best algorithms for lifestream data embeddings

Extracting embeddings from event sequences is a useful approach that can be used for learning users preferences, advising customers products based on their clickstream data and many other real-world applications.

However, classical methods for analyzing event sequences that are attributed to a person and capture their regular and routine behavior do not capture relationships between single object and its immediate neighborhood very well. Also, it usually requires a labeled data to operate this.

Therefore, new methods for data augmentation and event sequences extraction is needed.

Related work

CoLES: Contrastive Learning for Event Sequences with Self-Supervision

Dmitrii Babaev, AIRI, Sber AI Lab, Moscow, Russia
Nikita Ovsov, Ivan Kireev, Maria Ivanova, Sber AI Lab, Moscow, Russia
Gleb Gusev, Sber AI Lab, MIPT, Moscow, Russia
Ivan Nazarov, AIRI, Moscow, Russia
Alexander Tuzhilin, New York University, New York, USA

The paper on which our replication study is based on.

Event sequence metric learning

Dmitrii Babaev*, Sberbank AI Lab
Ivan Kireev, Sberbank AI Lab
Nikita Ovsov, Sberbank AI Lab
Mariya Ivanova, Sberbank AI Lab
Gleb Gusev, Sberbank AI Lab
Alexander Tuzhilin, New York University

Metric learning method for producing embeddings of complex event sequences. The main advantage is that almost no pre-processing is needed for complex even streams to get their compact embedding.

Attributed Sequence Embedding

Zhongfang Zhuang, Xiangnan Kong, Elke, Rundensteiner, Worcester Polytechnic Institute, {zzhuang, xkong, rundenst}@wpi.edu
Jihane Zouaoui, Aditya Arora, Amadeus IT Group, {jihane.zouaoui, aditya.arora}@amadeus.com

Study about extracting data embeddings from attributed sequences.

A Framework For Contrastive Self-Supervised Learning And Designing A New Approach

Technical Report

William Falcon, New York University, NY, Lightning Labs, NY, waf251@nyu.edu
Kyunghyun Cho, New York University, NY, kc119@nyu.edu

Study with analysis of contrastive learning approaches and conceptual framework for Contrastive Learning.

Project main goal

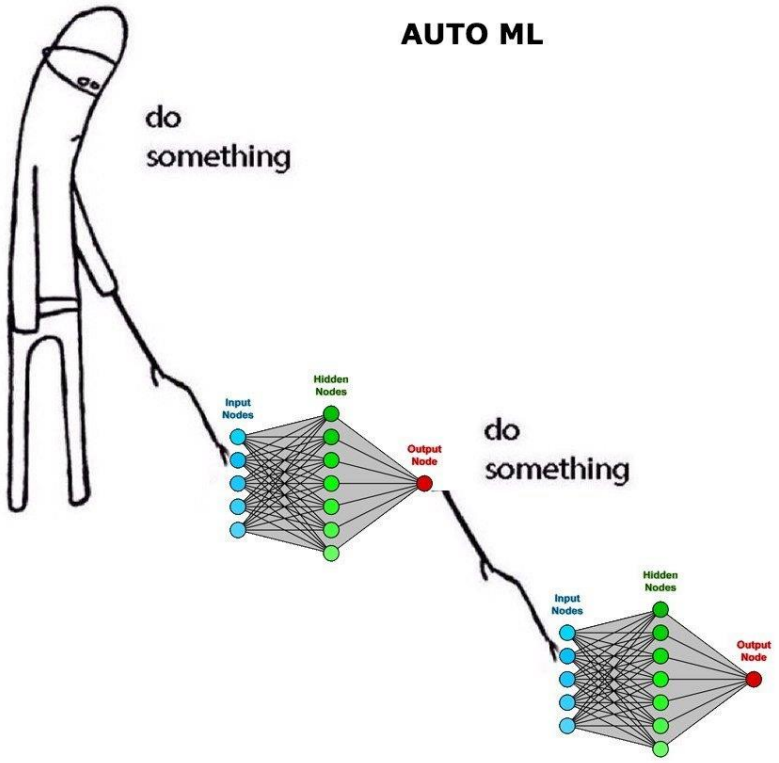
ClosedAI

Replicate the methods, proposed in the paper, on different types of event sequence data and compare its results.

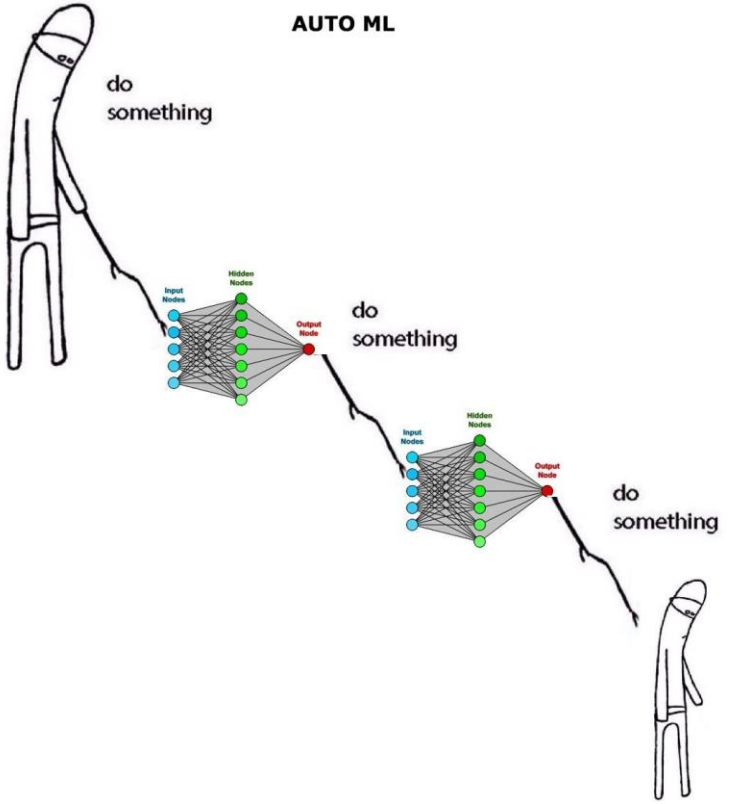
1. Get familiar with the pytorch-lifestream library
2. Run the proposed methods for learning embeddings on two types of data(transactions and clickstream)
3. Implement downstream machine learning task on both types of data and compare its results

Contrastive Self-Supervised Learning for Data Embeddings

How we thought it works



How we it actually works



Contrastive Self-Supervised Learning for Data Embeddings

Self-Supervised Learning (SSL) is one such methodology that can learn complex patterns from unlabeled data. Contrastive self-supervised learning (CSL) has been recently found successful for semi-supervised learning (SSL).
In CSL, the goal is to generate representations of instances such that similar instances are near each other and far from dissimilar ones. In supervised learning, associative labels determine the similarities among instances.

Different from conventional multidimensional data, the sequential data are not represented as feature vectors of continuous values, but as sequences of categorical items with variable-lengths. Examples of sequential data include click streams of web users, purchase histories of online customers, and DNA sequences of genes.
Sequential data usually requires a careful design of itsembedding before being fed to data mining algorithms. One of the feature learning problems on sequential data is called sequence embedding, where the goal is to transform a sequence into a fixed-length embedding.

CoLES framework

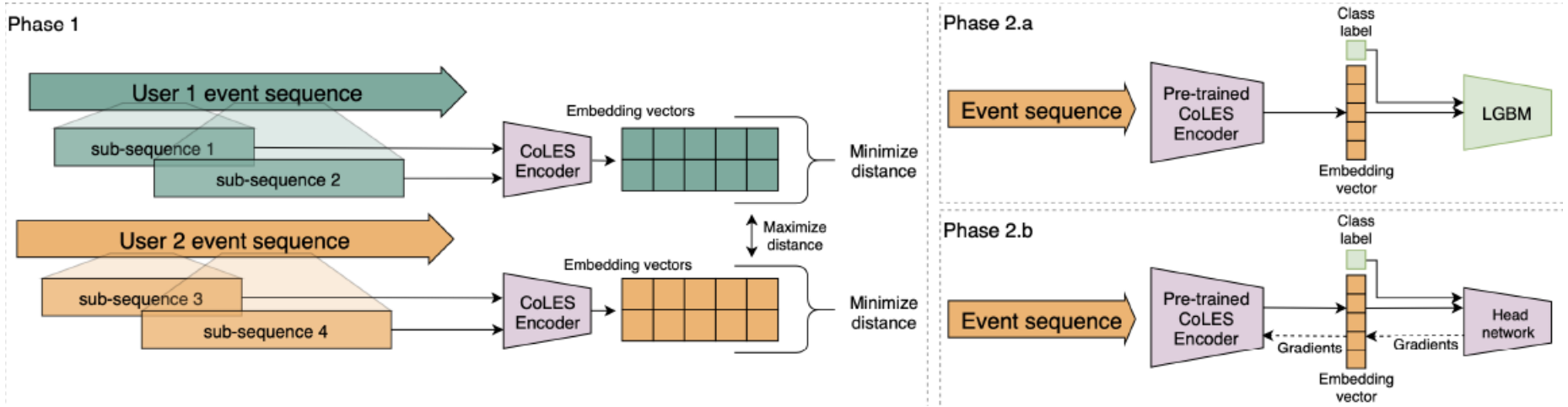


Figure 1: General framework. Phase 1: Self-supervised training. Phase 2.a Self-supervised embeddings as features for supervised model. Phase 2.b: Pre-trained encoder fine-tuning.

Our implementation consists only of **Phase 1** and **Phase 2.a** steps. Also we've implemented **aggregate baseline encoder** and **random feature encoder** on both datasets in order to investigate more data embeddings' extraction methods and to compare the resulting scores of these three methods.

Baseline methods

ClosedAI

Random feature encoder has architecture which is similar to CoLES, but it is not pre-trained. Therefore, all weights are random.

In **aggregate baseline encoder** categorical features are decomposed into One-Hot Encoder. Numerical features are decomposed according to the resulting columns. A separate encoder is used to generate features. The encoder interface is the same as that of RNN CoLES. Aggregate baseline encoder doesn't take into account the sequential nature of data, unlike CoLES.

General pipeline

ClosedAI

Tuning

Configuring the model, choosing hyper-parameters, tuning the model and obtaining embeddings

Metrics

Achieve metrics for further comparison

We combine information about each user by implementing ptls library

Preprocessing

Tune CatBoost using embeddings from previous step to predict whether a client has a higher education

Down-stream task

DataFusion 2022 datasets

ClosedAI

We have datasets with industrial data of transactions and clickstreams.
Due to this data we need to predict whether a client has a higher education.

transactions
19821910 rows

	user_id	mcc_code	currency_rk	transaction_amt	transaction_dttm
0	000932580e...	5411	48	-361.07230	2020-08-03 08:05:23
1	000932580e...	5499	48	-137.31398	2020-08-05 01:27:40
2	000932580e...	5499	48	-138.84981	2020-08-05 03:28:11
3	000932580e...	4829	48	-309.47653	2020-08-06 00:36:29
4	000932580e...	5411	48	-133.47370	2020-08-09 00:30:13

clickstream
126752515 rows

	user_id	cat_id	timestamp	new_uid
0	000143baebad4467a23b98c918ccda19	165	2021-01-30 20:08:12	1873448
1	000143baebad4467a23b98c918ccda19	165	2021-01-31 20:06:29	1873448
2	000143baebad4467a23b98c918ccda19	308	2021-01-31 20:12:00	1873448
3	000143baebad4467a23b98c918ccda19	931	2021-01-31 22:12:00	1873448
4	000143baebad4467a23b98c918ccda19	931	2021-02-01 16:57:00	1873448

target
8509 rows

	bank	higher_education
0	3755b59782464456bac1aec1a44e0db3	0.0
1	604a550439d644718ea6e1693fbf03dc	0.0
2	542d4776ebe5454fb8ab36f1c276fe0e	1.0
3	ee37fecea44d475ca030cde7ff7d545d	0.0
4	18617a1100f44a99b3a0772341fec3db	0.0

Transactions preprocessing

ClosedAI

transactions.csv -> pd.read_csv('transactions.csv') ->

	user_id	event_time	mcc_code	currency_rk	transaction_amt
0	00093...	tensor(1596,...	tensor(1, 2, 2, 13, ...	tensor(1, 1,...	tensor(-361.0723,...
1	0009...	tensor(1596,...	tensor(1, 10, 26, 26, ...	tensor(1, 1,...	tensor(-44.5269, ...
2	000b29...	tensor(1596,...	tensor(3, 21, 1, 3, ...	tensor(1, 1,...	tensor(-334.5844, ...
3	000cd...	tensor(1596,...	tensor(74, 1, 1, 3, ...	tensor(1, 1,...	tensor(-1923.6536, ...
4	000e0d...	tensor(1596,...	tensor(6, 1, 1, 11, ...	tensor(1, 1,...	tensor(-728.3863, ...

Clickstreams preprocessing

clickstreams.csv -> pd.read_csv('clickstreams.csv') ->  -> PySpark -> 

	user_id	cat_id	event_time
0	018d951f...	[29, 1, 1, 60, 60, 60, 60, 24, 24, 24, 3, 3, 3...	[1611903458, 1612164353,...
1	01ef0e6...	[4, 2, 2, 3, 23, 3, 3, 23, 23, 3, 3, 3, 3, 3, ...	[1610973443, 1612144860,...
2	0570e0...	[12, 8, 11, 36, 36, 36, 12, 12, 31, 1, 7, 1, 3...	[1612968600, 1613295720...
3	0677be1...	[12, 12, 12, 12, 7, 12, 2, 2, 2, 1, 12, 12, 2,...	[1622604120, 1622936168,...
4	0914be...	[1, 1, 9, 9, 9, 31, 33, 1, 9, 9, 9, 31, 9, 9, ...	[1612744599, 1612756008...

Hyper-parameters

ClosedAI

Table 1: Hyper-parameters for CoLES training

Model	CoLES
Embedding size	256
Loss function	Contrastive loss
Learning rate	0.001
Resampling	Sample Slices
Number of splits	5
Size of subsequence	25-200
Encoder	GRU

We’ve tried different parameters on both transactions and clickstream datasets, but the parameters listed in that are the best for models training.

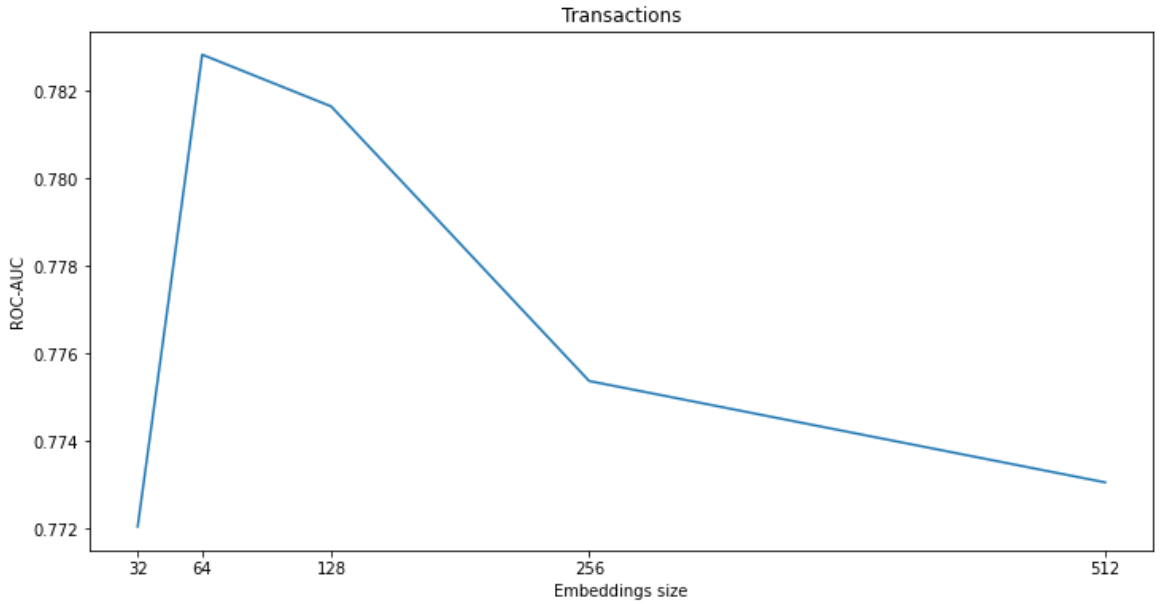
Results

Table 2: Scores on transactions and clickstream datasets with different metrics

	CoLES		**Agg baseline**	**Random encoder**
Metrics	Transaction dataset	Clickstream dataset	Transaction dataset	Transaction dataset
F1 score	0.8615	0.8334	0.8522	0.8519
ROC AUC score	0.7583	0.5936	0.5014	0.5047
Precision	0.9605	1	1	1
Accuracy	0.7708	0.7145	0.7427	0.7421

Results

- Transaction dataset gives better result than Clickstream dataset
- CoLES performs better than other used models (agg baseline and random encoder)



Embedding dimensionality vs Quality for transactions

Contribution

ClosedAI

Denis Grankin

Code

Mikhail Konenkov

Report
Presentaion

Irena Gureeva

Report
Presentaion

Ivan Apanasevich

Code

Ekaterina Andreichuk

Code

Conclusion

- Pytorch-lifestream library was implemented for event sequences data extraction and CoLES method was used for embeddings
- Data Fusion 2022 datasets were investigated and prepared for testing
- Downstream tasks were implemented for data embeddings extracted with CoLES, Agg Baseline and Random Encoder
- Results were obtained for different methods and datasets, their performance was compared