Adaptation of contrastive learning techniques for event sequences to a predictive task

Li Jianghai
t-li@edu.hse.ru

ABSTRACT

This study explores the application of contrastive learning methods on event sequences for predictive tasks. Specifically, we evaluate Contrastive Predictive Coding (CPC), TABNET, and SAINT algorithms across various scenarios related to event sequence prediction. Our results demonstrate that SAINT consistently outperforms CPC and TABNET in both unsupervised learned embeddings and supervised finetuned encoder evaluations. SAINT's utilization of self-attention and intersample attention mechanisms proves effective for modeling complex event sequences, achieving superior predictive performance. This research contributes to advancing the understanding and application of contrastive learning in event sequence analysis, highlighting its potential for enhancing predictive modeling capabilities in diverse domains.

Key words: Event Sequences; Contrastive Learning; Predictive Task

1. INTRODUCTION

In image recognition, contrastive learning methods have strong potential in both training and recognition. Through self-supervised learning, contrastive learning and other methods, useful representations can be learned from some relatively large-scale data without manual labeling. This paper aims to explore the application effects of different contrastive learning methods on bank transaction data, especially on three specific data sets. These datasets contain various customer transaction records and corresponding labels, providing rich experimental scenarios.

Across the bank's many transactional businesses, we treat customer transaction behavior as an important piece of information, such as fraud detection, risk assessment and customer segmentation. But traditional supervised learning methods rely on large amounts of labeled data, and obtaining this data is often costly and time-consuming. In contrast, contrastive learning appears to be more advantageous. This approach can improve the performance of downstream tasks by designing appropriate pre-training tasks to learn effective features on unlabeled data.

This study focuses on validation on bank datasets using different contrastive learning methods. We will use more than three contrastive learning algorithms, including but not limited to contrastive predictive coding (CPC), TABNET, and SAINT, and apply these methods to the scenario_age_pred, scenario_chun_pred, and scenario_bowl2019 data sets. Our goal is to evaluate the advantages and disadvantages of these methods when dealing with different types of transaction records by comparing the accuracy of predictions on these data.

We hope to make the following contributions through final experiments and analyses:
1. Find the optimal contrastive learning algorithm and verify its performance on financial transaction data.
2. Explore the effectiveness of contrastive learning in enhancing downstream classification and prediction tasks under unlabeled data conditions.
3. Analyze relevant real data to provide reference for future continuous research.

The experimental results of this article will demonstrate the analysis and processing of different contrastive learning on different data, and later provide customer behavior prediction methods for the financial services industry. We believe that these research results will help us better understand the role of contrastive learning in real application scenarios and increase the data agility of relevant institutions.

2. RESEARCH BACKGROUND

In banking scenarios, how to learn effective representations from event sequences is very important. The CoLES method developed by Sber AI Lab excels at this task and outperforms other methods when solving downstream tasks. CoLES focuses on describing the current state of an object through pre-training tasks and then generating a high-quality representation. However, this method also has some limitations, especially that it fails to fully consider the dynamic changes of the sequence and does not train the model to predict the future state of the object. These limitations are a bottleneck for prediction-centric tasks.

In addition, LANET (Label-wise Attention Network) is a self-attention network based on the Transformer architecture. It can be used to specifically handle multi-label classification tasks of event sequences. The architecture is characterized by calculating self-attention between label views, which can effectively capture the complex relationships between labels. Therefore, this method is particularly suitable for bank transaction data, by analyzing the complex dependencies between transaction records and tags. If we utilize LANET, we can more effectively capture the time dependencies and label associations between transaction data, thereby improving classification performance.

On the other hand, some other methods such as Contrastive Predictive Coding (CPC) solve the prediction task, but they are often not that effective when dealing with sequence data. That is, there is a lack of methods that can combine sequence representation learning and prediction capabilities. So we need to know more about other methods.

SimCLR: SimCLR is a self-supervised learning method based on image data that learns representations through data augmentation and contrastive loss. SimCLR was originally designed for use with images, but it can also be used with time series and transactional data.

BYOL (Bootstrap Your Own Latent): BYOL differs from other contrastive learning methods in that it does not require negative samples. A mutual learning approach between two different views of the BYOL network can improve prediction capabilities. And they are particularly useful for processing transaction data with high similarity.

MoCo (Momentum Contrast): MoCo enables efficient learning by maintaining a dynamic dictionary and comparing the data represented in the dictionary. MoCo is designed to handle large and dynamically changing data sets.

SWAV (Swap Allocation Between Views): SWAV implements self-supervised learning by exchanging allocations between different views. It performs well when handling diverse data and is particularly suitable for highly heterogeneous data.

3. DATASET

This paper involves multiple datasets to verify the effect and performance of the model in different scenarios. The following important datasets: Scenario Age Prediction, Scenario Churn Prediction, and Scenario Bowl 2019 datasets. These data sets are all from Sberbank financial datasets.

The Scenario Age Prediction dataset is a dataset comprising 44 million anonymized credit card transactions from 50,000 individuals was utilized to predict age groups. The multiclass target labels are available for 30,000 records, with a balanced distribution within this subset. Each transaction entry includes date, type, and amount charged.

The Scenario Churn Prediction dataset is a dataset consisting of 1 million anonymized card transactions from 10,000 clients was employed to predict churn probability. Each transaction includes date, type, amount, and Merchant Category Code. The binary target labels are available for 5,000 clients, with labels being nearly balanced.

Finally, **The Scenario Bowl 2019** , which contains a wealth of market-related data. Integration and utilization of these datasets, such as the application at https://github.com/dllllb/ptls-experiments/tree/main . It verifies the effectiveness of various models such as the CPC model.

4. COMPARISON LEARNING METHODS

1. Contrastive Learning in Computer Vision (CPC)

Contrastive Learning has made significant progress in the field of computer vision, inspired by large-scale pre-trained models of text. Contrastive Learning solves the problem of how to design pretext tasks and loss functions for unsupervised learning to improve representation learning without solving specific supervised tasks. This makes unsupervised learning an important stepping stone towards robust and general representation learning.

The basic assumption of contrastive learning is that the context of related values often conditionally relies on the same shared higher-order latent information. By framing this as a prediction problem, the model automatically infers these features of interest for representation learning.

Specific methods include:

1. Mapping high-dimensional features to low-dimensional space: This mapping makes conditional prediction simpler and more effective, and helps the model learn more discriminative feature representations.
2. Use powerful autoregressive (AR) models to predict multiple steps into the future: With this approach, models are better able to capture complex dependencies in time series data.
3. Rely on Noise-Contrastive Estimation (NCE) to calculate the loss: similar to the loss calculation method of language models, NCE achieves effective feature learning through contrastive learning.

A common strategy in contrastive learning is to learn the differences and similarities between different samples in an image or feature space by maximizing the similarity of positive sample pairs and minimizing the similarity of negative sample pairs. This method shows excellent performance on tasks such as image classification, object detection and semantic segmentation. In addition, it also shows great potential in cross-domain learning, domain adaptation and semi-supervised learning.

Through contrastive learning, the model can learn more discriminative and generalizable feature representations without explicit labels. This makes it one of the important technologies that have attracted much attention in deep learning research and practical applications.

Goal: Maximize Mutual Information (MI)

$$I(x;c) = \sum_{x,c} p(x,c) log \frac{p(x,c)}{p(x)p(c)}$$

c: context

Then,

$$I(x;c) = \sum_{x,c} p(x,c) \log \frac{p(x|c)}{p(x)}.$$

Next, we will introduce information entropy.

Information content measures the information brought about by a specific event, while entropy is the expectation of the amount of information that may be generated before the result is released. The formula for the amount of information is as follows,

$$h(x) = -log_2 p(x)$$

Information entropy is the expectation of the amount of information brought by all possible events considering all possible values of the random variable.

$$H(x) = -\sum p(x) log_2 p(x)$$

Information entropy can also be used as a measure of system complexity. If the system is more complex and the more types of different situations occur, then its information entropy is relatively large.

The concept of mutual information is familiar to everyone. It is based on Shannon entropy and measures the degree of dependence between two random variables. Unlike ordinary similarity measurement methods, mutual information can capture the nonlinear statistical correlation between variables, so it can be considered to measure the true dependence.

$$\begin{aligned} I(x;c) &= \sum_{x,c} p(x,c) \log \frac{p(x \mid c)}{p(x)} \\ &= \sum_{x,c} p(x,c) \log p(x \mid c) - \sum_{x,c} p(x,c) \log p(x) \\ &= \sum_{c} p(c) \sum_{x} p(x \mid c) \log p(x \mid c) - \sum_{x} \sum_{c} p(x,c) \log p(x) \\ &= -H(x \mid c) + H(x) \\ &= H(x) - H(x \mid c) \end{aligned}$$

That is to say, the amount of x entropy reduction due to the introduction of c, that is, the amount of x uncertainty reduction.

In addition, we also need to introduce Preliminary.

$$z_t = g_{enc}(x_t)$$
$$c_t = g_{ar}(z_{\leq t})$$

z_t: hidden representation
g_{enc}: input encoder
c_t: context
g_{ar}: context encoder

Among them, the density proportion function:

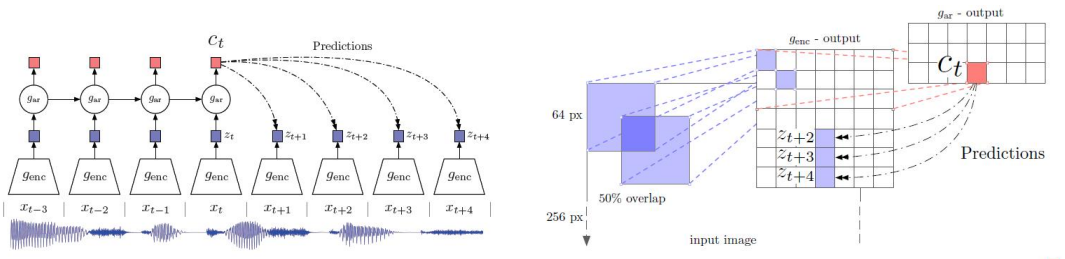
$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right),$$

Finally, we need to introduce InfoNCE Loss and Mutual Information Estimation:
Optimization goals of InfoNCE:

$$\mathcal{L}_{\mathrm{N}} = -\underset{X}{\mathbb{E}}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

$$\begin{aligned}
\mathcal{L}_{\mathrm{N}}^{\mathrm{opt}} &= -\underset{X}{\mathbb{E}} \log\left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}\right] \\
&= \underset{X}{\mathbb{E}} \log\left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} \sum_{x_j \in X_{\mathrm{neg}}} \frac{p(x_j|c_t)}{p(x_j)}\right] \\
&\approx \underset{X}{\mathbb{E}} \log\left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)}(N-1)\underset{x_j}{\mathbb{E}} \frac{p(x_j|c_t)}{p(x_j)}\right] \\
&= \underset{X}{\mathbb{E}} \log\left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)}(N-1)\right] \\
&\geq \underset{X}{\mathbb{E}} \log\left[\frac{p(x_{t+k})}{p(x_{t+k}|c_t)} N\right] \\
&= -I(x_{t+k}, c_t) + \log(N),
\end{aligned}$$

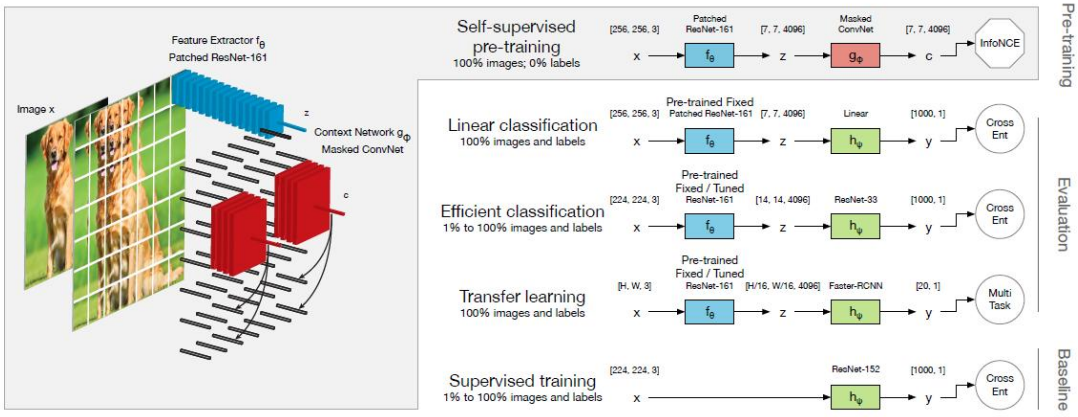
CPCv1 model structure:



The model structure of CPCv1 can be divided into the following main parts:

1. **Encoder:** The encoder is responsible for converting input data (such as images or audio) into a series of latent representations. For image data, a commonly used encoder is a convolutional neural network (CNN); for audio data, a recurrent neural network (RNN) or a convolutional network may be used.
2. **Context Network:** The context network is used to integrate previous latent representations and generate a context vector. In CPCv1, this network is usually a recurrent neural network (RNN) such as LSTM or GRU. Context vectors capture local and global information of the input data.
3. **Contrastive Loss:** The core idea of CPCv1 is to maximize the mutual information between future potential representations and context vectors through contrastive learning. Specifically, the model attempts to predict future potential representations and compares them to the current context vector through a linear classifier. Contrastive loss (usually using Noise-Contrastive Estimation, NCE) is used to distinguish between positive pairs (real future representations) and negative pairs (randomly selected other representations).
4. **Latent Representation Prediction:** During training, the model needs to predict potential representations multiple time steps into the future. Through this multi-step prediction, the model not only learns information at the current moment, but also captures the time series dependence of the data. This part consists of a linear transformation and the vector output by the context network.

CPCv2 model structure:



CPCv2 introduces stronger data enhancement and regularization techniques to improve the generalization ability of the model. Data augmentation techniques can generate more diverse training samples, while regularization techniques can prevent models from overfitting, allowing them to perform well on different tasks and data sets.

Specific method

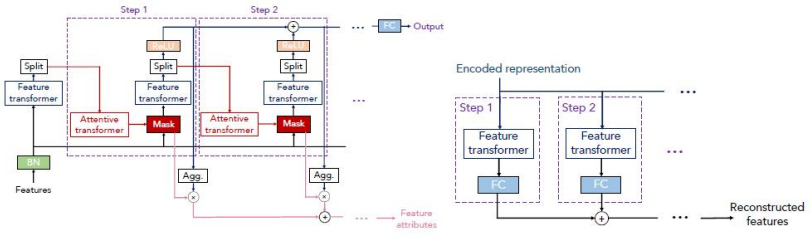
1. Contrastive Prediction of Latent Representations: CPCv2 continues to adopt the contrastive learning method to maximize the mutual information between context vectors and future potential representations by predicting future potential representations. The model compares the current context vector with the future representation through a linear classifier and calculates a contrastive loss (NCE).
2. Multi-Step Prediction: Similar to CPCv1, CPCv2 also performs multi-step prediction, but in each step of prediction, CPCv2 introduces a more complex architecture and optimization technology to make the prediction more accurate and stable.

2. Tabnet

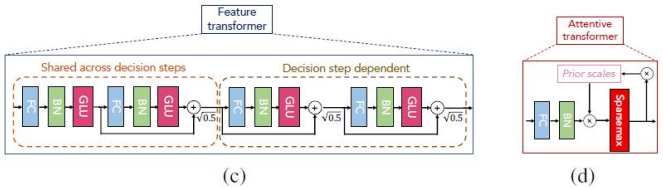
XGBoost and LightGBM have dominated tabular data processing in recent years. They are based on gradient boosting decision trees (GBDT), not DNN (deep neutral network). DNN has not made much progress in processing tabular data.

GBDT occupies a dominant position in tabular data processing mainly because it is efficient and performs well when processing approximate hyperplane boundaries. It is also highly interpretable and easy to explain afterwards. The TabNet method provides an end-to-end learning framework that does not require feature processing, uses gradient descent for training, and selects features for reasoning through the sequential attention mechanism in each step of the decision tree, with instance-level interpretability. TabNet is not only comparable to or better than other table learning methods in performance, but also provides two kinds of interpretability: local interpretability shows the importance and combination of input features, and global interpretability quantifies the contribution of each input feature. In addition, TabNet introduces unsupervised pre-training to predict masked features, which is the first self-supervised learning method for tabular data.

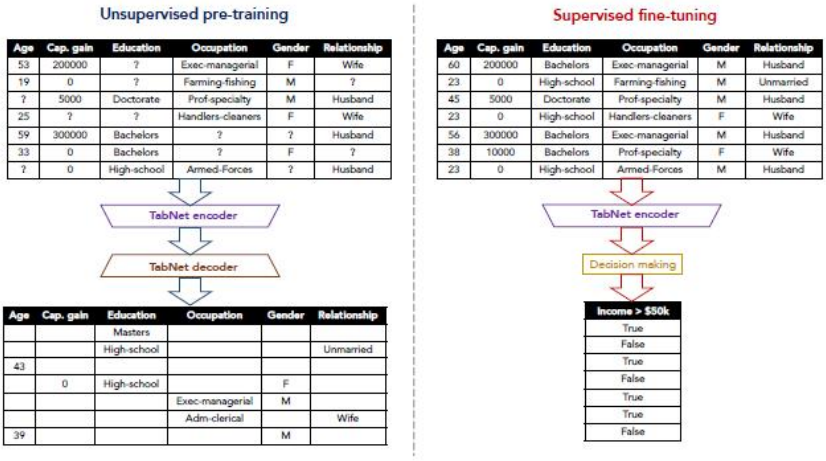
Tabnet model structure:



(a) TabNet encoder architecture (b) TabNet decoder architecture



(c) (d)



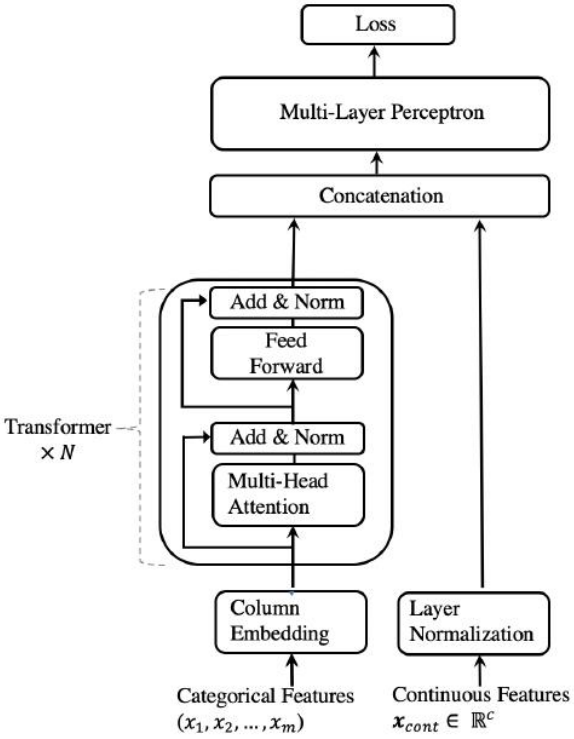
1. **Input Layer:**
TabNet directly accepts raw tabular data as input, without feature engineering or preprocessing. This enables the model to flexibly handle a variety of tabular data.
2. **Decision Steps:**
TabNet processes data through a series of decision-making steps. At each decision step, the model selects features for further processing based on the information from the previous step. Each decision step consists of the following subcomponents:
3. **Feature Transformation:**
The feature transformation layer uses a fully connected layer and Batch Normalization to map the input data into a high-dimensional feature space.
4. **Mask Generator:**
The mask generator utilizes the sequential attention mechanism to generate a selection mask that determines which features are used in the current decision step. The mask generator outputs a mask matrix that selects features by multiplying them with the input features.
5. **Feature Selection Mechanism:**
TabNet uses the Sequential Attention mechanism to select features for reasoning at each decision step. This mechanism allows the model to select different features in different instances, thereby achieving instance-wise feature selection.
6. **Decision Outputs:**
At each decision step, the model produces a decision output that is accumulated to form the final prediction. The decision output is generated through a fully connected layer and ReLU activation function.

TabNet achieves flexible, interpretable and powerful tabular data processing capabilities by combining the sequential attention mechanism and deep learning framework. It can directly process raw tabular data, perform instance-level feature selection, and provide detailed interpretability analysis, while using unsupervised pre-training to improve model performance.

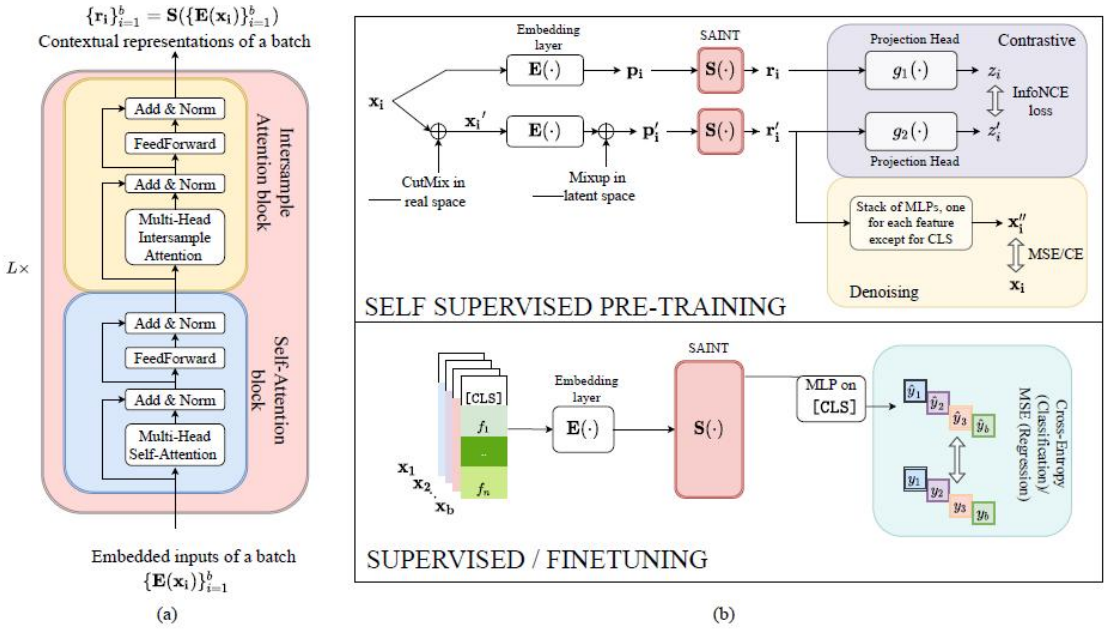
3. Self-Attention and Intersample Attention Transformer (SAINT)

Before that, we need to understand TabTransformer. Among them, the tree model problem: cannot handle streaming data and multi-modal data, Semi-supervised learning is not possible, SOTA's missing value or noise processing techniques cannot be used. MLP question: The model or representation has no interpretability, Lack of robustness to missing values or noise, Semi-supervised learning is not possible.
Therefore, a transformer-based tabular data model was proposed to solve the problems of these two models.

The TabTransformer model treats tabular data as a hyperstructured subset of language, where each row corresponds to a "sentence" and each column value corresponds to a "word" or token. Unlike conventional language models, "sentences" in tabular data are of fixed length, and the order of "words" is not important but is agreed upon. At each position, the "word" value is a fixed categorical feature, rather than being freely selected from all words. In this way, TabTransformer can effectively learn the relationships and dependencies between columns to predict and classify tabular data.



Therefore, based on this, we propose a new model **(SAINT) .**



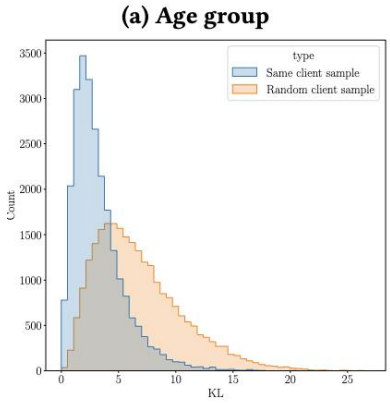
The SAINT model combines the self-attention mechanism and the inter-sample attention mechanism to make it more effective when processing tabular data. The self-attention mechanism (Self-Attention) enables the model to capture the complex relationships between columns in the table, while the inter-sample attention mechanism (Intersample Attention) allows the model to establish connections between different samples, thereby leveraging more context. information to make more accurate predictions. The combination of this dual attention mechanism makes SAINT perform well in the classification and prediction tasks of processing tabular data. Compared with the TabTransformer model, it has stronger generalization ability and higher accuracy.

5. EXPERIMENTAL RESULTS ANALYSIS

Because the provided data set is a type of tabular data. In addition, we need to encode events one by one to adapt to various models later (**The event encoder and The sequence encoder**).

DATASET ANALYSIS

Next, we need to analyze the data. For example,



Periodicity and repeatability of the data. KL-divergence between event types of two random sub-sequences from the same sequence is compared with KL-divergence between sub-sequences of different sequences.

DATASET SPLIT

The dataset was split into training and test sets. Specifically, 10% of individuals from the labeled data were reserved as the test set for evaluation purposes. The remaining 90% of labeled data, along with unlabeled data, formed the training set used for model learning. Hyperparameters for each method were selected using random search through 5-fold cross-validation on the training set.

For semi-supervised/self-supervised techniques, all transactions from the training sets, including unlabeled data, were utilized during training. In contrast, the unlabeled portions of the datasets were disregarded when training supervised models.

Baseline: LightGBM for Tabular Data

LightGBM, a powerful Gradient Boosting Machine (GBM), is widely recognized as a robust baseline for tabular data with diverse features. In our experiments, LightGBM is employed as the primary model for downstream tasks. We explore two types of alternative input features:

1. Hand-crafted aggregate features derived from raw transactional data. Numerical attributes such as amount undergo aggregation functions (e.g., sum, mean, std, min, max) across all transactions per sequence. For instance, aggregating 'sum' over the numerical field 'amount' yields the feature 'sum of all transaction amounts per sequence'. Categorical attributes, such as merchant category (MCC) code and transaction type, are grouped by each unique value, and aggregation functions like

count, mean, and std are applied over all numerical attributes. For example, applying 'mean' to the numerical attribute 'amount' grouped by 'MCC code' results in the feature 'mean amount of all transactions for the specific MCC code' per sequence.
2. The embedding of transaction sequences produced by an encoder network. This encoder model is trained in a self-supervised fashion.
These approaches leverage LightGBM's strengths in handling tabular data with heterogeneous features, showcasing its effectiveness in predictive modeling tasks.

RESULT

Unsupervised learned embeddings with LightGBM model downstream evaluations:

Dataset	Age group Accuracy	Churn AUROC	Assess Accuracy
Baseline	0.629 ± 0.002	**0.827 ± 0.010**	0.592 ± 0.004
cpc1_embeddings	0.614 ± 0.006	0.801 ± 0.013	**0.602 ± 0.003**
coles_transformer	**0.646 ± 0.003**		

Supervised finetuned encoder with MLP head evaluation:

Dataset	Age group Accuracy	Churn AUROC	Assess Accuracy
rtd_finetuning	0.622 ± 0.003	0.791 ± 0.016	0.571 ± 0.003
cpc_finetuning	**0.625 ± 0.005**	**0.804 ± 0.017**	**0.594 ± 0.002**

Comparison of encoder types:

Dataset	Age group Accuracy	Churn AUROC	Assess Accuracy
LSTM	0.621	**0.823**	**0.620**
TABNET	0.604	0.812	0.612
Transformer	0.622	0.780	0.542
SAINT	**0.635**	0.818	0.601

6. CONCLUSION

In this study, we evaluated three algorithms—Contrastive Predictive Coding (CPC), TABNET, and SAINT—across multiple evaluation scenarios. SAINT consistently demonstrated superior performance compared to CPC and TABNET in both unsupervised learned embeddings and supervised finetuned encoder evaluations. SAINT's utilization of self-attention and intersample attention mechanisms proved highly effective for tasks involving complex sequence data, highlighting its robustness and versatility in predictive modeling.