Learning Transactions Representations for Information Management in Banks: Mastering Local, Global, and External Knowledge

Alexandra Bazarova[a,1], Maria Kovaleva[a,1], Ilya Kuleshov[a,1,*], Evgenia Romanenkova[a,1], Alexander Stepikin[a,1], Aleksandr Yugay[a,1], Dzhambulat Mollaev[b], Ivan Kireev[b], Andrey Savchenko[b], Alexey Zaytsev[a]

[a]*Skolkovo Institute of Science and Technology (Skoltech), Bolshoy Boulevard, 30 p.1, Moscow, 121205, , Russia*
[b]*Sber AI Lab, Kutuzovsky Avenue, 32, Moscow, 121165, , Russia*

Abstract

In today's world, banks use artificial intelligence to optimize diverse business processes, aiming to improve customer experience. Most of the customer-related tasks can be categorized into two groups: 1) local ones, which focus on a client's current state, such as transaction forecasting, and 2) global ones, which consider the general customer behaviour, e.g., predicting successful loan repayment. Unfortunately, maintaining separate models for each task is costly. Therefore, to better facilitate information management, we compared eight state-of-the-art unsupervised methods on 11 tasks in search for a one-size-fits-all solution. Contrastive self-supervised learning methods were demonstrated to excel at global problems, while generative techniques were superior at local tasks. We also introduced a novel approach, which enriches the client's representation by incorporating external information gathered from other clients. Our method outperforms classical models, boosting accuracy by up to 20%.

Keywords: representation learning, deep learning, financial transactional data, external context

*Corresponding Author
Email address: i.kuleshov@skoltech.ru (Ilya Kuleshov)
[1]Equal contribution

1. Introduction

The banking industry must keep up with current trends to remain competitive. Technological advancements such as Artificial Intelligence (AI) have significantly transformed how banks process and analyze data, allowing them to extract valuable insights for more informed decision-making and effectively address complex challenges [1, 2].

Among various types of data collected and managed by banks, AI-based analysis of financial transactions appears to be the most promising for various operational and strategic purposes [3, 4]. Such solutions already play a crucial role in risk assessment, improving the accuracy of credit scoring, advanced bankruptcy prediction, and financial distress identification [5]. They also help to automate fraud detection, identify transaction anomalies, and protect both banks and their customers [6]. Moreover, AI-driven customer segmentation and personalized marketing strategies allow banks to tailor services to individual customer needs, enhancing customer satisfaction and loyalty [3]. While these approaches enable banks to optimize operations, reduce costs, and manage risks, they often depend on task-specific methodologies, large labeled datasets, and expert-defined features, limiting their usage.

All these challenges may be addressed by representation learning via neural networks, a powerful solution that benefits from the vast amounts of transaction history sequences stored by modern banks. It focuses on extracting hidden patterns from data, capturing intrinsic features into low-dimensional embeddings. These representations can further be utilized in various downstream tasks, offering a more scalable and adaptable approach to employing data for business-oriented goals [7, 8, 9, 10, 11]. This general pipeline of SSL applications in the banking industry is further discussed in Section 3.

From a business perspective, representations of the customers' transaction sequences are supposed to reflect data properties at both global and local levels. We define *global properties* as a characteristic of a customer's transaction history as a whole. In this case, high-quality global representations should be helpful when comparing and classifying the sequences as entities [8, 12]. In contrast, the state of a customer at a specific point in time is described by *local properties*. Local representations are used for tasks associated with momentary customer behavior, such as next-event prediction [13] or credit card fraud detection [14]. Moreover, local information should reflect the *dynamics* of clients' lives and be useful for change point detection [15, 16]. However, the methods in the literature typically focus on either global or local embedding

properties, completely ignoring representation dynamics.

Another drawback of existing models is that they do not consider the external context when creating an embedding for each customer. While choosing macroeconomic indicators requires expert knowledge, such information may be represented by the common characteristics of the other customers' transactions, which were shown to be strongly correlated [17, 18]. Thus, a reasonable approach is to use different aggregations of the representations of all customers to form the external context vector. Previous works ignored this concept or designed very narrow methods, significantly increasing the number of the model parameters size [19]. However, accounting for this information may potentially boost the quality of the models [15].

To conclude, this work develops a representation learning method that considers local and global data patterns in transactional data. Additionally, the paper explores ways of incorporating the external context into the obtained representations. To solve these tasks, we extend existing strategies of working with sequential data and adopt the most promising techniques. More particular, the main contributions of the paper are the following:

1. We propose several techniques based on generative self-supervised learning to obtain transactional data representations with strong local and global properties. Our methods offer a good trade-off between these properties across various scenarios, providing the most versatile representations for transaction data at the moment;
2. We suggest an efficient procedure that utilizes the attention mechanism to involve the external context when constructing data representations, further enhancing the model quality for most applied problems;
3. We provide an extensive pipeline for evaluating the informativeness of data representations in terms of domain-specific properties that suit various needs. Our pipeline includes four diverse, open datasets, four downstream problems for them, and one common task. We also validate the models on an industrial-scale private dataset. The source code to run our benchmark and reproduce all experiments is publicly available[2].

The rest of the article is organized as follows. The next section discusses related research on representation learning in the transactional data domain. Section 3 describes the proposed methods and the datasets used. In Sections

[2] `https://github.com/romanenkova95/transactions_gen_models`

4 and 5, we present the results of our experiments, which evaluated the global and local properties of the data. Section 6 contains our approach to considering the external context. Sections 6 and 7 are dedicated to discussing our results and conclusions.

2. Related works

This section briefly describes the existing self-supervised representation learning methods for transactional data. Generally, there are two SSL paradigms: contrastive and generative. Below, we describe the most relevant models among these two approaches. We start the section by mentioning the supervised models relevant to the banking domain. Finally, we accompany the review with the Temporal Point Process (TPP) modeling methods explicitly designed for the analysis of event sequences — the type of data transaction histories belong to.

2.1. Supervised methods for representation learning

Solutions in banking often lean towards traditional paradigms incorporating expert domain knowledge [3]. In credit scoring, researchers prefer the classical machine learning methods [20] since they tend to outperform deep learning approaches on top of manually constructed features [21, 22, 23, 24]. Credit card fraud detection methods analysis provides similar insights [25, 6, 26, 27].

On the other hand, recent papers support applying supervised deep learning methods to raw transactional data, modifying approaches to account for the particularities of the target domain. Thus, when working with sequential data, modern works consider recurrent neural networks (RNNs) and Transformers [15, 28, 29, 30]. Other notable solutions include adjusting for the time since the last transaction [31] and working with small subsequences to curb the vanishing gradients problem [32].

2.2. Self-Supervised Learning Paradigm

Lately, there has been a surge in attention to self-supervised methods (SSL). Such methods try to extract information from unlabeled data. This allows researchers to skip the costly gathering of expert annotations, leveraging the large bodies of low-cost raw data. The SSL methods can be roughly divided into contrastive and generative approaches [33].

Contrastive approaches have been gaining popularity in financial transactions [7, 9, 34]. All contrastive methods aim to yield close representations of "similar" objects and distant representations of "dissimilar" ones. The way one measures similarity may be very simple, such as the one in the papers [7, 35, 11], which contrasts between slices from different sequences. More complex options include combining distance and angle-based metrics [34] and training with hierarchical losses [36].

Generative models use various training approaches to learn the hidden data distribution. The resulting knowledge may then be utilized to produce plausible-looking data. These approaches often originate from Neural Language Processing (NLP) [37, 38]. Autoencoder is a very popular generative method due to its simplicity and efficiency in different modalities [39]. The article [10] employs the variational autoencoder to learn bank customer representations, which is useful for downstream tasks [5]. Masked language models (MLMs) take the autoencoder idea one step further. Such models are trained via restoring randomly changed tokens and perform well on diverse benchmarks [37, 40].

According to the latest research, generative methods outperform contrastive ones at missing value prediction, among other tasks [12]. This fact may be indicative of better local properties. On the other hand, while supposedly good locally, generative models are known to be less efficient at global tasks. Here, contrastive models, like CoLES [7], come into play because they typically consider a sequence of events as a whole, enforcing the embeddings' global properties. As no evident leader exists, we explore both approaches for transactional data, testing the resulting representations for the desired properties.

2.3. Temporal Point Process Models

Financial transactional data can be treated as event sequences, differing from conventional time series and natural language data in some key aspects:

- Observations in time series appear uniformly in time, which is not valid for transactional sequences [35]. In turn, text is not tied to time at all.
- In contrast to time series datasets, the sequences in a transactional dataset may be of different lengths.

- Transactions have heterogeneous features, e.g., MCC codes are categorical, and transaction amounts are continuous. Generally, neither the time series nor the NLP data exhibit this peculiarity.

Event sequence problems are commonly solved using TPP models. Following this approach, one generally aims to reconstruct conditional event intensity, determining the type and frequency of future events based on the history of earlier observations [41, 42]. Standard objectives include likelihood maximization, time-to-next-event (return time) estimation, and next-event-type prediction [43, 13].

Modern research incorporates neural networks for the intensity function parametrization. For example, the Neural Hawkes Process (NHP) [43] applies the ideas of the classic Hawkes Process [41] to RNN-generated latent embeddings. The methods from papers [44, 33, 45] are based on the same idea but stick to the architectures with the attention mechanism [46]. Another class of neural TPP models includes recently proposed continuous convolutional networks (CCNNs) [47, 48]. This architecture successfully handles the irregularity of event sequences in many cases. For example, the COTIC [13] approach uses a deep CCNN backbone model and two multilayer perceptron heads on top of it for return time and event type prediction.

Note that TPP models are not typically studied in terms of their embedding quality, which is the goal of this research. However, the models' natural problem statement [43, 13] may enhance highly efficient representations of event sequences regarding their local properties. Consequently, we add several TPP models to our benchmarking.

3. Methodology

The general pipeline for incorporating self-supervised learning in the banking industry is illustrated by Figure 1. The general idea is to pretrain an *encoder model* on large open corpuses of unlabelled data, producing informative embeddings for transactional sequences. These representations can then be used for solving downstream tasks. Further information on the specific approaches to learning such embeddings can be found in Section 4. In the remainder of this section, we describe our approach to evaluating the resulting representations.

As outlined in Section 1, different business tasks in banking often rely on different aspects of embeddings. In this work, we study the trade-off between

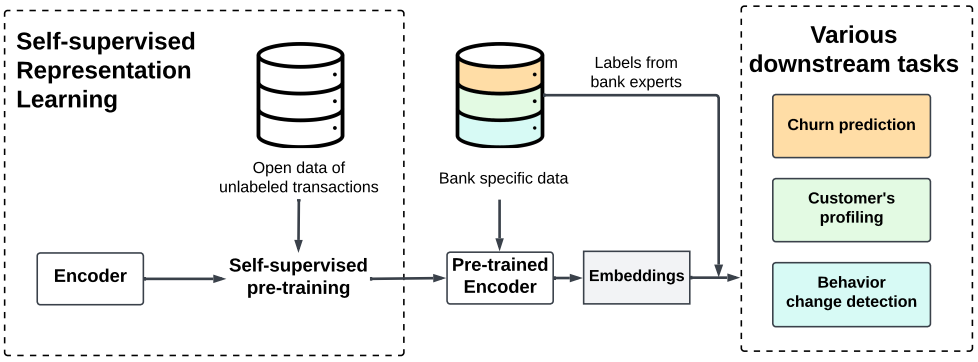

Figure 1: The general flow of the application of self-supervised learning to industrial tasks in banking.

local and global properties of transactional data representations obtained using various models. *Global properties* characterize the client's behaviour throughout the entire history of his/her transactions. In contrast, *local properties* show the nature of the client's current state and its evolution through time.

3.1. Global properties' validation methods

To assess the quality of the obtained representations regarding their global properties, we follow the paper [7], which proposed the CoLES method. More precisely, following the approach illustrated in Figure 1, we evaluate the obtained embeddings on several downstream tasks. This requires each dataset used in this work to have a classification problem behind it. For example, the *Churn* dataset contains binary targets: for each bank customer, we aim to identify whether he left the bank (see Section 5 for complete information about considered data).

To sum up, the global evaluation pipeline consists of three steps:

1. For an initial sequence of transactions of length T_i related to the i-th user, we obtain the *global representation* $\boldsymbol{H}^i \in \mathbb{R}^d$, which characterizes the transaction history as a whole. In the case of sequence-to-sequence encoder architectures, a pooling operation is applied to the encoded sequence to create a single representation of an entire series.
2. Given representation $\boldsymbol{H}^i$, we predict the target label $y_i \in \{0, 1, \ldots, C\}$ using gradient boosting. We stick to the LightGBM [49] model applied

in the paper [7]. It works fast enough for large data samples and provides sufficiently high-quality results. The hyperparameters of the boosting model are fixed and do not vary across all the base models under study.

3. Classification results are evaluated using a standard set of metrics described in 3.4.

3.2. Local properties validation methods

We suggest using a set of techniques to evaluate the quality of the obtained representations in terms of their local properties.

Local downstream targets. By analogy with the global downstream tasks discussed above, business problems frequently require local clients' information to solve them. We propose considering such local downstream classification tasks for the Churn, Default, and HSBC datasets (see Section 5).

In the *Churn* dataset, original target labels indicate clients who eventually stopped using the bank's services. It is reasonable to assume that the behaviour of customers who are about to leave a bank begins to change in advance. For example, they make fewer transactions and stop topping up their card balances. Given this, all transactions by the "churn" user i within a month before his/her departure are tagged with $c_j^i = 1$, while the others are marked with $c_j^i = 0$. We select an "early outflow" horizon of one month for empirical reasons.

The *Default* dataset also follows a similar logic. Here, we create local binary labels that reflect the client's transition to the "pre-default" state, noting that the default of a client typically corresponds to three missing monthly instalment payments.

HSBC initially contains local binary labels for each transaction record, indicating whether it is malicious. In the case of this dataset, we do the opposite and add global targets that show if a client has ever been a victim of bank fraud.

As a result, we propose to evaluate the quality of local representations by predicting a local binary target c_j^i. The corresponding classification tasks are solved using a two-layer perceptron. The better local embedding describes the client's current behaviour, the better a simple classifier model can solve applied problems.

Next transaction MCC prediction. Next-event-type prediction is a common objective in TPP modelling [43, 44, 13]. Formally, it implies solving the multiclass classification problem: given a "frozen" local representation $\boldsymbol{h}^i_{t_j}$, a separate model predicts the following MCC (event type) for the i-th client. To achieve this, we train an additional, simple two-layer perceptron.

It is important to consider that there are a lot of rare MCCs in the datasets. From a business perspective, such categories are often less meaningful. Therefore, to simplify the task, we kept only the transactions with the top-100 most popular codes.

Challenge in inferring local representations. Generally, not all the studied models are efficient in a sequence-to-sequence mode. That is, they may be unable to produce representations for a specific timestamp in a given sequence. Thus, we use a sliding window of size w to obtain the local representations in all cases.

In particular, for the i-th user at time $t_j \in [t_w, T_i]$, we consider the subsequence $\boldsymbol{S}^i_{j-w:j}$. Here, T_i denotes the length of transactional history for this client. Next, the interval $\boldsymbol{S}^i_{j-w:j}$ is passed through the encoder model to obtain a *local representation* $\boldsymbol{h}^i_{t_j} \in \mathbb{R}^d$. This approach is illustrated in Figure 2.

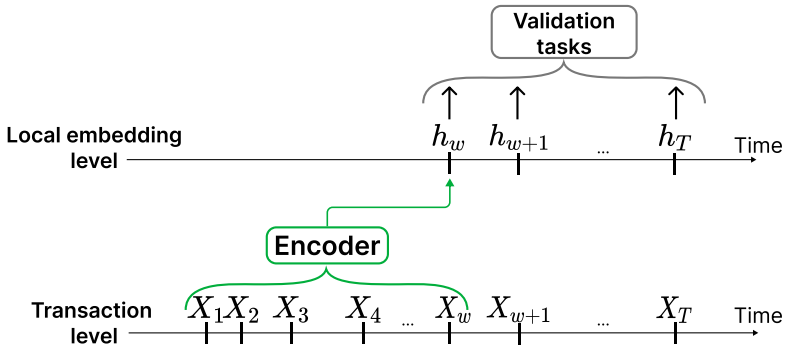

Figure 2: Scheme of the sliding window approach for local validation.

On the one hand, this technique helps us create local embeddings for any given model. Moreover, it enhances local representation properties, as we do not consider out-of-date information. On the other hand, a tiny window size may be insufficient to capture the broader patterns of a customer at a given time step. Additionally, the process does not permit obtaining local embeddings for times $t \leq t_w$. However, this issue should be relatively insignificant

in the case of long transaction sequences. Therefore, w becomes an additional hyperparameter that must be tuned to balance the above trade-off.

3.3. Strategy for validation of embeddings dynamism

An alternative approach for assessing the quality of the local representations is to evaluate their *dynamic properties*, i.e., the ability to reflect changes in the raw data. In this regard, we considered the change point detection (CPD) task [50, 51, 52] posed in the space of local representations.

Change points are time moments with an abrupt shift in sequential data distribution that may indicate transitions between states in the underlying process. If we consider the behaviour of bank customers as such a process, an example of a change point could be taking a loan or getting a new job. The task of change point detection is to identify the moments of such shifts accurately.

Let $\boldsymbol{H} = \{\boldsymbol{h}_{t_j}\}_{j=1}^{N} \subset \mathcal{H} \subset \mathbb{R}^d$ be the sequence of the client's local representations $\boldsymbol{h}_{t_j}$, where t_j is the timestamp of j-th transaction, $t_1 \leq \cdots \leq t_N$. It is natural to assume that if there is a change point in a time series at a particular moment τ, then, for any $t_i < t_j < \tau < t_k$, the following holds with high probability:

$$d(\boldsymbol{h}_{t_i}, \boldsymbol{h}_{t_j}) < d(\boldsymbol{h}_{t_i}, \boldsymbol{h}_{t_k}),$$

where $d(\cdot, \cdot)$ is a metric assigned to $\mathcal{H}$. Following [16, 51], we select the cosine similarity as our distance measure d. This metric is used in a series of experiments to illustrate how the representations evolve in the vicinity of a change point. To identify the change points in the embedding sequences, we employ the standard Dynp [52] method from the ruptures package[3].

3.4. Validation metrics

We utilize a set of classical metrics in our benchmark. As mentioned above, our validation techniques involve binary and multiclass classification problems, as well as change point detection tasks.

Classification metrics. For classification problems, we utilize the standard metrics: Accuracy, ROC-AUC, and PR-AUC [22, 23, 11]. For these three metrics, a better model produces higher values.

[3] `https://centre-borelli.github.io/ruptures-docs/`

Accuracy is defined as the ratio of correct predictions to all predictions made:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{1}$$

where TP — true positive, TN — true negative, FP — false positive, and FN — false negative predictions.

ROC-AUC, or the area under the Receiver Operating Characteristic (ROC) curve, represents the relationship between true positive rate (TPR or Recall) and false positive rate (FPR) for different thresholds. Similarly, the PR-AUC is the area under the Precision-Recall curve, which describes the connection between Precision and Recall (TPR). Formally,

$$\text{TPR} = \frac{TP}{TP + FN}, \quad \text{FPR} = \frac{FP}{TP + FN}, \quad \text{Precision} = \frac{TP}{TP + FP}. \tag{2}$$

To generalize to multiple classes, we use micro-averaging for Accuracy and weighted averaging for ROC-AUC and PR-AUC. This means that the contribution of each class to the overall result is proportional to the number of its representatives. Consequently, the result is insensitive to class imbalance.

Change point detection quality. We also evaluate the dynamics of the obtained embeddings via change point detection methods. Sticking to this field's best practices, we use the following CPD quality metrics: average detection delay (DD) [51] and detection accuracy [50] with different values of the margin parameter m.

DD is the average difference between the estimated and actual change points:

$$\text{DD} = \frac{1}{T} \sum_{i=1}^{T} (\hat{\tau}_i - \tau_i) I[\hat{\tau}_i \geq \tau_i], \tag{3}$$

where τ_i is the true change point, $\hat{\tau}_i$ is the predicted one, T is the total number of disorders and $I[\cdot]$ is the indicator function.

We assume that there is only one disruption per sequence. The detection accuracy A_m determines how often the detected change falls within the m-the neighbourhood of the true change point:

$$\text{A}_m = \frac{1}{T} \sum_{i=1}^{T} I[|\hat{\tau}_i - \tau_i| \leq m]. \tag{4}$$

An efficient change point detector has low detection delay and high detection accuracy for different margins m.

4. Models

Below, we describe specific encoder models and their modifications: supervised baselines in subsection 4.1, contrastive methods CoLES and TS2Vec in subsection 4.2, generative models such as autoencoders and autoregressive model in subsection 4.3, and TPP models in subsection 4.4. Finally, subsection 4.5 concludes with a description of our novel methods for obtaining and using the external context.

4.1. Supervised baselines

Obtaining an accurate markup for a real-world dataset is expensive and time-consuming. Nevertheless, supervised methods remain popular first-step solutions for representation learning problems [20, 6, 27]. We, in turn, focus on self-supervised methods in our work, renowned for their capacity to uncover latent data patterns from unlabeled data. To provide a deep understanding of the possible gap between embeddings from these approaches, we train several baseline models that adopt convolutional and recurrent neural network architectures.

In the experiments, we show the results for the model with the best performance on downstream tasks (see Section 3). The selected architecture consists of a single-layer Gated Recurrent Unit (GRU) [53] as the backbone, complemented by a multilayer perceptron (MLP) serving as the classification head. The architecture resembles the ET-RNN model and CoLES encoder [7, 28]. The aggregated hidden states of the backbone are used as inputs for the classification head during the training phase and as embeddings in the validation tasks. As an objective function, we employ the cross-entropy loss.

4.2. Contrastive models for transaction sequences

CoLES method. As a starting point in our study of SSL methods, we chose CoLES [7], a representation learning method for discrete event sequences. Below is a brief overview of this approach; see the original paper for more details.

As all contrastive SSL approaches [54], the CoLES model requires sets of positive and negative pairs for training. The authors of the method studied various ways to obtain such pairs. Eventually, they opted for the split strategy that considers two transaction subsequences from the same client to be a positive pair and from different clients — to be a negative one. This strategy does not disrupt the transactions' order, preserving their temporal structure.

Representations of the subsequences were obtained via a Long Short-Term Memory network [55] (LSTM) as an encoder model, which was trained using a contrastive loss [56]. In this work, we follow the same training pipeline as the original article.

TS2Vec model. The contrastive model TS2Vec [36] has emerged as one of the leading approaches for obtaining representations of classical multivariate time series data. TS2Vec stands out for its distinct positive and negative sample selection strategy and hierarchical aggregation to facilitate representation learning at different scales.

The authors have introduced contextual consistency, a novel approach for generating positive pairs. This approach designates representations at the same timestamp within two augmented contexts as positive pairs. Concerning negative samples, selection occurs in both instance-wise and temporal dimensions, complementing each other.

In the initial step, the described contrastive approach is employed on timestamp-level representations. Following this, max pooling is executed along the temporal axis. Subsequently, the contrastive loss is applied to the acquired representations, and this entire process iteratively continues for each level until instance-level representations are attained. This mechanism proficiently empowers the model to systematically capture contextual information at diverse resolutions within temporal data. In our study, we rely on the original implementation of this method (with several modifications for transactional data type).

Limitations of the considered contrastive approaches. The methods outlined above are only partially suited for modeling event sequences and thus have limitations.

The split strategy employed in CoLES explicitly encourages the model to build representations describing the user as a whole rather than capturing his local state at a specific moment. Moreover, the original paper only evaluated the global properties of the obtained embeddings since the users' classification was the primary downstream task.

TS2Vec, in turn, was primarily designed for standard time series samples with a uniform step. Therefore, it might be unfit for transactional data featuring irregular time intervals between observations [57]. In our work, we explore the adaptability of TS2Vec to transaction history representation learning and incorporate it as a baseline model for comparative analysis.

4.3. Our approaches to generative modeling for transaction sequences

Classic autoencoder model. The autoencoder (AE) [39] is a popular choice for generative models. Following this approach, the sequence is first encoded into a low-dimensional representation and then restored from it. The overall process is depicted in Figure 3. Both the encoder and the decoder networks are represented by LSTMs [55], so the backbone model architecture is identical to CoLES. We add two linear heads to the decoder for amounts and MCC prediction.

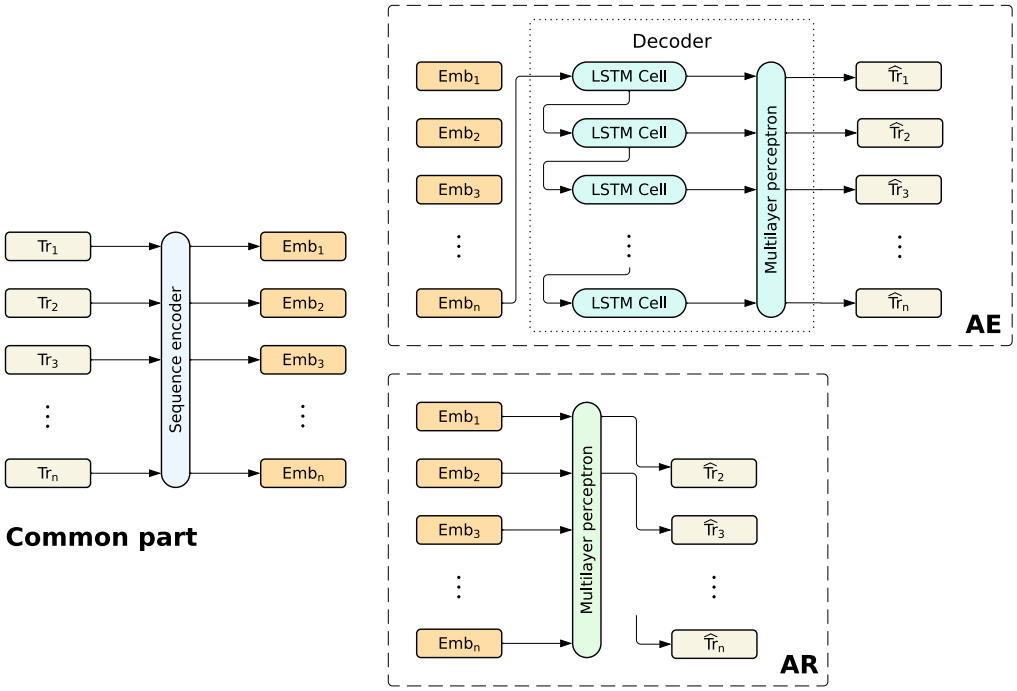

Figure 3: Scemes of the Autoencoder (AE) and Autoregressive (AR) methods.

Since transactional records contain both categorical (MCCs) and continuous (amounts) information, the reconstruction loss is divided into two parts. We use cross-entropy for the categorical features and mean squared error for the continuous features. The final loss function is a weighted sum of the two intermediate ones, with the weights being hyperparameters.

We found that the preprocessing of transaction amounts defined below is essential for the model to train successfully:

$$f(a) = \text{sign}(a) \ln(1 + |a|). \tag{5}$$

On the one hand, this transformation allows us to balance out the loss functions for amounts and MCCs without assigning them drastically different weights. On the other, it better reflects the nature of human perception: we tend to focus on the order of magnitude, ignoring the precise value.

For self-supervised approaches, it is very important to select the optimal complexity for the training objective [37, 40]. In our case, it turns out that the prediction of rare MCC codes is an unbearable problem. So, to simplify the training objective, the number of unique MCC codes is reduced to 100 for all datasets by clipping all less frequent MCC codes.

Masked language model. Diving deeper into generative methods, we also explore the masked language model (MLM), closely related to transformers [37, 58, 46]. The main components of transformers are self-attention and feed-forward blocks.

An MLM works by reconstructing randomly masked tokens. The method is depicted in Figure 4. Before feeding the sequence to our model, we randomly select 10% of the incoming tokens and preprocess them similarly to [37]. In particular, out of these 10%, we:

- mask 80%, swapping out the MCC code for a special token and zeroing out the transaction amount;
- swap 10% for another random transaction;
- and keep as is the remaining 10%.

Like in the AE model, we use multiple linear layers to acquire the predictions from the encoder embeddings. The loss function is also a combination of cross-entropy and mean-squared error, but here, it is calculated only for the 10% subset we sampled above. We also clip our MCC codes to 100 ones and preprocess the amounts like for the AE model.

It should also be noted that all transformer embeddings have equal receptive fields, each covering the whole sequence (which is not correct for RNN embeddings we use in other methods). This implies that, in essence, any of these representations can be used in the downstream and validation tasks as representations of all the given tokens. Following previous work on transformers, we choose the first token's embedding since that is traditionally where the *CLS* token is.

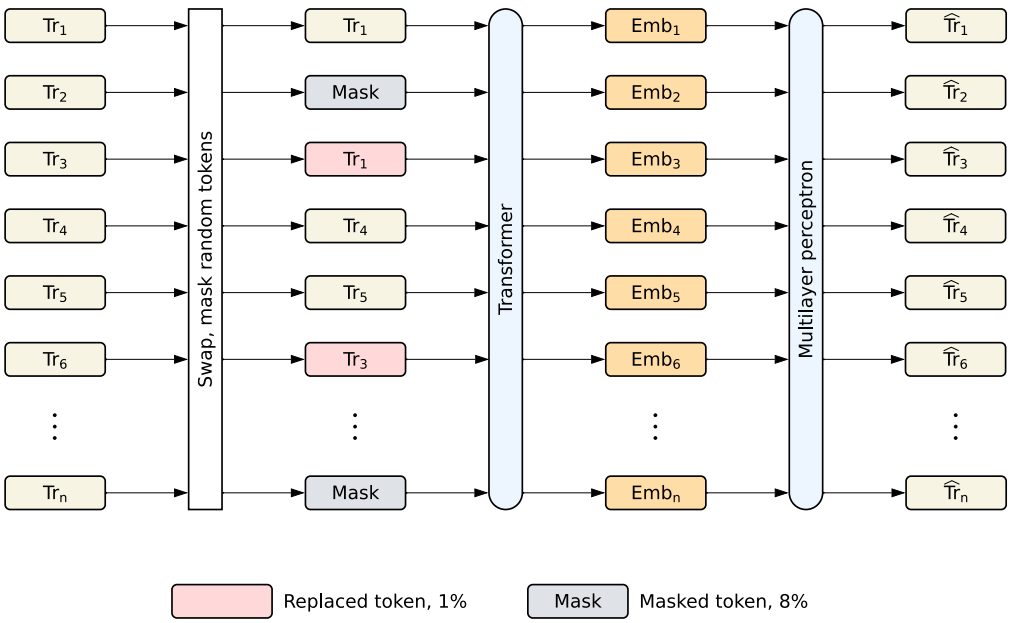

Figure 4: Scheme of the Masked Language Model (MLM) method.

Autoregressive approach. Autoregressive (AR) modeling is extensively used in CV [59, 60, 61, 62], as well as in sequential domains like NLP [63, 64, 65] and Audio [66, 67]. Such models aim to predict the next item in a sequence. In contrast to token embeddings in autoencoders, those in autoregressive models do not use information regarding future tokens. This characteristic empowers autoregressive models to capture more intricate patterns, as demonstrated by their superior performance in text generation tasks [68].

In our experiments, we employ a recurrent neural network similar to the CoLES model. We utilize the model's last hidden state to represent the transaction sequence since it encapsulates complete information about the entire sequence. The model is trained to predict the next transaction, encompassing the MCC code and transaction amount, as illustrated in Figure 3. In terms of transaction preprocessing and the loss function, they resemble the techniques employed in autoencoders, outlined in Section 4.3.

4.4. Temporal Point Processing models

When working with irregular time series or event sequences, it is crucial to consider that the records (measurements, indicators) have distinct time

steps between each other. Such a problem statement is standard for TPP modeling.

In this work, we choose three state-of-the-art models as baselines of that type: the Neural Hawkes Process (NHP) [43], the Attentive Neural Hawkes Process (A-NHP) [44], and the COTIC [13] model. All these approaches follow a similar logic: they utilize deep neural networks for conditional event intensity parametrization. The main optimization objective here is the log-likelihood of the observed event sequences. The models' architectures and peculiarities are briefly described below.

Neural Hawkes and Attentive Neural Hawkes Process. The NHP [43] model originates from the Hawkes Process [41] — one of the classic ways of event sequence modeling. Its key concept is self-excitation, i.e., past events temporarily raise the probability of future events. This impact is assumed to be positive, additive, and exponentially decaying over time. NHP generalizes the assumption by determining the conditional event intensities from a hidden state of a recurrent neural network. The authors propose a novel continuous-time LSTM [55] architecture that allows querying event representations at any time, even if no actual event occurred at that moment.

The A-NHP approach [44] develops the same ideas and suggests replacing the original recurrent encoder with a simpler parallelizable attention-based architecture [46]. The paper proves such a solution is more efficient while it provides comparable or better validation results.

COTIC model. Continuous 1D convolutional neural networks (CCNN) have also demonstrated their efficiency for TPP modeling [47, 48]. A recently proposed COTIC approach [13] utilizes a deep CCNN to obtain event sequence representations and parameterize the intensity function of the underlying process. On top of it, COTIC applies two multi-layer perceptron heads for return time and event type prediction. Notably, these heads are trained jointly with the core part of the model using task-specific loss functions. This strategy may enforce the desired local properties of the embeddings obtained from the encoder. However, using only an encoder from the COTIC, without additional heads, might lead to a degradation in quality. We refer interested readers to the paper itself [13] for more details on the architecture and the losses.

4.5. Usage of external information based on local representations of transactional data

According to the current research [15, 7], it is beneficial to consider the macro context around users. We propose to construct an external context representation vector by properly aggregating the embeddings obtained from all or some selected users. The procedure is presented in Figure 5 and is described as follows:

1. We build local representations for all users and each unique transaction moment.
2. For each client, we select all local embeddings from the dataset close to the current time point but before it.
3. Obtained vectors are aggregated in the resulting vector of the external context representation.

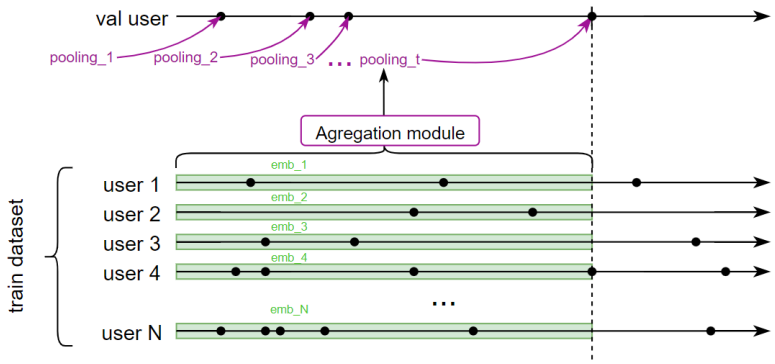

Figure 5: General pipeline for obtaining external context vectors.

To validate the ability to improve the representations via external information, we concatenate each user's local embeddings with this resulting vector and follow our general validation pipeline (see Section 3 for details).

In the experiments with the external context aggregation, we use CoLES as an encoder for building local representations. However, potentially, any model that transforms a sequence of transactions into a sequence of their representations fits our pipeline.

Classical methods of context vector aggregation. Averaging and maximization were used as classical aggregation methods. Our motivation comes from an analogy with applying Mean and Max Pooling operations in convolutional neural networks [69]. In the first case, we obtain the external context vector by component-wise averaging the local vectors for all users in the stored dataset, and in the second case, we take the maximum value for each component. Like in convolutional networks, such aggregation methods are designed to generalize the environment for the current point.

Methods based on the attention mechanism. Following a fruitful idea from the Recommender Systems domain, we suppose that some bank clients may behave more similarly than others. Thus, such people determine the user's closest environment and help to describe his behavior better. The aggregation methods described above do not consider the users' representation relationships and, as a consequence, their similarity. To overcome this drawback, we propose applying the attention mechanism. It was initially designed to account for interword similarities during context aggregation in the machine translation task [46] that resembles our problem.

In this work, we utilize two variants of the attention mechanism: without a learnable attention matrix and with it. The corresponding form for the external context vectors for a given point in time are:

$$\boldsymbol{B}_t = X\sigma(X^T\mathbf{h}_t), \quad \boldsymbol{B}_t = X\sigma(X^TA\mathbf{h}_t), \tag{6}$$

where $\mathbf{h}_t \in \mathbb{R}^m$ is a vector of local representation for the considered client; $X \in \mathbb{R}^{m\times n}$ is a matrix, which columns are embeddings of all n users (except considered) at a given time; $A \in \mathbb{R}^{m\times m}$ is a trainable matrix; and σ denotes the softmax function $\sigma(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_{j=1}^{d}\exp(z_j)}$.

In the first case, the user similarity metric is the softmax of the scalar product. In the second, vectors of representations from the dataset are additionally passed through a trained linear layer. We train the attention matrix via the CoLES model training pipeline (see Section 4.2 for more information).

5. Data overview

In this study, we compare our methods using public and private datasets of transaction records from different banks' customers. We briefly describe the data below and provide the main technical characteristics of open datasets in Table 1.

Churn. The Churn dataset was initially dedicated to identifying potential customers about to leave. We also use this problem for our global validation (Section 3). Due to the nature of such events, the target value distribution is unbalanced. For the local validation tasks, we assume that clients change their behavior one month before their last transaction.

Default. Default has a resembling structure to the Churn problem. The corresponding global downstream task is to classify clients as able to repay the bank loan. This task is close to the real-world business problem of credit scoring. Likewise, we mark transactions before the default month as "pre-default" for local validation. It is also important to note that this dataset has a significant class imbalance, which may lead to increased standard deviations of the metrics.

HSBC. The HSBC competition proposes the fraud identification task. The original dataset contains local targets indicating if each transaction was made by the client personally or if it was malicious. We also add client-level global tags showing that at least one operation in a transaction history sequence was fake. Obviously, there are far fewer labels indicating that transactions are fraudulent, so the target variable is highly imbalanced.

Age. The Age dataset implies predicting bank clients' age group based on their transactions history. Four groups are in total, creating a relatively balanced multi-class global downstream task. The Age differs from other considered datasets. Its samples do not include exact timestamps for each transaction. Instead, they only have a serial number of the day the transaction is made starting from some base date. In addition, there is no natural local task for this data (since the global target's dynamic is negligible in the considered period), so we skip this local downstream validation step.

Real-world private dataset. A major bank has provided us with a proprietary dataset containing the anonymized data of a subset of real clients. The total number of unique customers in this data is five million, with a time range of approximately one year. This dataset includes characteristics such as each client's age and gender, which we use for both binary and multi-class classification during the global validation process. Similarly to Age, we skip local downstream task validation and evaluate only performance on the MCC prediction.

Table 1: Basic statistics of the open datasets used in the experiments.

	Churn	Default	HSBC	Age
# of transactions	490K	2M	234K	26M
# of clients	5K	7K	4K	30K
Min transaction length	1	300	1	700
Max transaction length	784	300	3467	1150
Median transaction length	83	300	40	863
# of MCCs	344	309	307	202
# of global classes	2	2	2	4
Class balance	yes	no	no	yes

Change point detection datasets. CPD experiments are conducted on both real and synthetic data. For the real-world scenario, we use a subset of 35 clients from the Churn dataset containing long transaction sequences with a currency change serving as the change point. Since the real data with annotated change points is limited, we also consider synthetic sequences. For this purpose, we constructed samples by combining subsequences from different clients.

Data preprocessing. We carry out minimal preprocessing for analyzed datasets. Our steps include bringing all timestamps to one format, removing unnecessary columns, and encoding MCCs by frequency. Based on the above logic (Subsection 3.2), we also add local binary targets for each of the three datasets: Churn, Default, and HSBC.

To correctly evaluate our models' performance, we split the data into three parts: training set (80% of all users), validation set (10%), and test set (10%). In our pipeline, the encoder models use only the training set for learning. Regarding validation, additional local classifiers (MLP heads) use the same training/validation/test split as the encoders. In turn, global validation classifiers (gradient boosting) are trained on training and validation sets and evaluated on the test set.

6. Results

Below, we provide the results we achieved during our work. The implications of each figure and table are discussed meticulously in the corresponding text. This section concludes with an executive summary, which contains a ranking of the considered models, as well as a brief analysis of said ranking.

Within the framework of our project, we compare several classes of transactional data models described in Section 4. Contrastive self-supervised approaches include CoLES [7] and TS2Vec [36]; generative self-supervised methods are presented by the classic autoencoder (AE) (Subsection 4.3), the masked language model (MLM) (Subsection 4.3), and the autoregressive model (AR) (Subsection 4.3); and Temporal Point Process baselines encompass the NHP [43], the A-NHP [44], and the COTIC [13] models. Additionally, we include the results for the best supervised baseline (Best sup.), which was trained using the global markup. It is important to note that these models could be superior to the self-supervised ones in many cases, as they directly use the correct labels.

The details on the models' architecture, implementation, and hyperparameters are given in Appendix A. We release the GitHub repository[4] containing the reproducible Python code for all the experiments conducted in this research.

6.1. Analysis of local and global properties of representations

The main results for the four open datasets, emphasizing the trade-off between local and global properties of the models, are illustrated in Figure 6. More details are provided in Tables 2.

In these graphs, the x-axis represents the global benchmark, while the y-axis represents the local one (predicting the next MCC code). Thus, the closer the dot is to the upper right corner, the better the corresponding model.

Note that the two datasets under consideration (Default and HSBC) are highly imbalanced. Consequently, the accuracy is not a relevant classification metric in this case. Thus, we suggest that the integral metrics (ROC-AUC and PR-AUC) are the main quality measures in the experiments.

The results (see Tables 5,6,7 for a summary) demonstrate that the generative models (AE, MLM, and AR) outperform the contrastive methods (CoLES and TS2Vec) at local validation tasks, aligning with our expectations and the generally accepted understanding of generative models [12]. Importantly, they also typically keep up with the contrastive methods or even exceed the state-of-the-art CoLES approach on global tasks.

The more detailed conclusions are the following:

[4] `https://github.com/romanenkova95/transactions_gen_models`

Table 2: Quality metrics for global and local embedding validation results. All metrics in the Table should be maximized. The results are averaged by three runs and are given in the format $mean \pm std$. The best values are **highlighted**, and the second best values are underlined. "Best sup." stands for the best supervised baseline model. Note that there are no local binary targets for the Age dataset.

Method	Global validation			Local validation: next MCC			Local validation: binary target		
	Accuracy	ROC-AUC	PR-AUC	Accuracy	ROC-AUC	PR-AUC	Accuracy	ROC-AUC	PR-AUC
Churn									
Best sup.	0.67±0.02	0.71±0.02	0.75±0.02	0.25±0.01	0.64±0.01	0.17±0.00	0.73±0.01	0.55±0.01	0.31±0.01
CoLES	0.67±0.02	0.73±0.02	0.77±0.03	0.23±0.00	0.64±0.01	0.16±0.01	**0.73**±0.00	0.56±0.01	0.32±0.01
TS2Vec	0.65±0.02	0.69±0.01	0.75±0.02	0.16±0.01	0.63±0.01	0.14±0.01	0.65±0.05	0.52±0.03	0.28±0.02
AE	0.69±0.01	**0.76**±0.01	**0.82**±0.01	0.27±0.01	0.70±0.00	0.21±0.00	**0.73**±0.00	0.57±0.00	0.33±0.02
MLM	0.67±0.01	0.75±0.01	0.81±0.02	**0.28**±0.00	0.72±0.00	0.21±0.01	**0.73**±0.00	0.57±0.01	0.33±0.01
AR	**0.70**±0.00	0.75±0.01	0.77±0.01	**0.28**±0.00	**0.73**±0.00	**0.23**±0.00	**0.73**±0.00	**0.58**±0.01	0.34±0.01
NHP	0.68±0.02	0.75±0.02	0.77±0.01	0.24±0.00	0.54±0.02	0.13±0.01	0.63±0.00	0.54±0.04	0.40±0.04
A-NHP	0.67±0.02	0.72±0.01	0.75±0.01	0.21±0.01	0.56±0.01	0.13±0.01	0.64±0.00	0.55±0.03	**0.41**±0.02
COTIC	0.67±0.03	0.73±0.03	0.75±0.02	0.22±0.03	0.51±0.01	0.12±0.00	0.63±0.00	0.49±0.01	0.36±0.01
Default									
Best sup.	0.91±0.00	0.53±0.04	0.09±0.03	0.31±0.00	0.66±0.00	0.21±0.01	0.90±0.01	0.50±0.01	0.01±0.00
CoLES	0.86±0.02	0.57±0.01	0.06±0.01	0.34±0.01	0.75±0.00	0.26±0.00	0.93±0.08	0.54±0.04	0.01±0.00
TS2Vec	**0.93**±0.00	0.55±0.03	**0.09**±0.02	0.25±0.02	0.66±0.01	0.18±0.01	**0.99**±0.00	**0.61**±0.05	0.01±0.00
AE	0.56±0.02	0.49±0.08	0.05±0.01	0.33±0.01	0.73±0.00	0.26±0.00	0.96±0.00	0.53±0.05	0.01±0.00
MLM	0.73±0.02	0.54±0.03	0.06±0.01	0.34±0.00	0.75±0.00	0.26±0.00	**0.99**±0.00	0.56±0.04	0.01±0.00
AR	0.87±0.01	0.56±0.07	0.08±0.02	**0.35**±0.00	**0.76**±0.00	**0.28**±0.00	0.97±0.02	0.44±0.05	0.01±0.00
NHP	0.92±0.01	0.55±0.01	0.07±0.02	0.30±0.00	0.60±0.02	0.15±0.01	**0.99**±0.00	0.52±0.04	0.01±0.00
A-NHP	0.72±0.02	0.48±0.05	0.04±0.01	0.29±0.01	0.60±0.01	0.16±0.00	0.98±0.01	0.60±0.06	0.01±0.00
COTIC	0.90±0.01	**0.58**±0.02	0.08±0.02	0.30±0.00	0.57±0.01	0.14±0.01	0.94±0.07	0.38±0.02	0.00±0.00
HSBC									
Best sup.	0.92±0.00	0.67±0.03	0.15±0.01	0.71±0.01	0.87±0.03	0.78±0.04	1.00±0.00	0.37±0.13	0.01±0.00
CoLES	0.92±0.00	**0.69**±0.04	0.15±0.02	0.69±0.02	0.90±0.01	0.82±0.01	**1.00**±0.00	0.38±0.09	0.01±0.00
TS2Vec	**0.95**±0.04	0.45±0.07	0.05±0.04	0.65±0.04	0.78±0.06	0.66±0.04	0.97±0.04	0.54±0.05	**0.05**±0.07
AE	0.92±0.00	0.66±0.04	0.14±0.03	0.72±0.04	0.91±0.01	0.84±0.02	**1.00**±0.00	0.44±0.09	0.01±0.00
MLM	0.93±0.00	**0.69**±0.02	0.15±0.02	**0.80**±0.01	**0.92**±0.00	**0.85**±0.00	**1.00**±0.00	**0.81**±0.02	0.02±0.00
AR	0.92±0.00	0.65±0.03	0.15±0.01	0.79±0.02	**0.92**±0.00	**0.85**±0.00	**1.00**±0.00	0.79±0.02	0.02±0.01
NHP	0.92±0.00	0.64±0.04	**0.19**±0.04	0.66±0.00	0.76±0.01	0.67±0.02	**1.00**±0.00	0.48±0.02	0.01±0.01
A-NHP	0.93±0.00	0.58±0.02	0.15±0.01	0.66±0.00	0.73±0.00	0.63±0.00	**1.00**±0.00	0.32±0.06	0.00±0.00
COTIC	0.92±0.00	0.55±0.03	0.10±0.02	0.65±0.01	0.68±0.05	0.64±0.01	0.96±0.02	0.51±0.04	0.01±0.00
Age									
Best sup.	0.61±0.00	0.85±0.00	0.65±0.00	0.33±0.00	0.67±0.00	0.22±0.00			
CoLES	**0.61**±0.00	**0.85**±0.00	**0.66**±0.00	0.32±0.00	0.71±0.00	0.24±0.00			
TS2Vec	0.60±0.01	0.84±0.00	0.64±0.00	0.18±0.02	0.61±0.01	0.17±0.01			
AE	0.52±0.02	0.78±0.01	0.54±0.02	0.35±0.00	0.72±0.01	0.26±0.00			
MLM	0.59±0.01	0.84±0.01	0.63±0.01	0.33±0.00	0.71±0.00	0.24±0.01		NA	
AR	0.59±0.01	0.83±0.00	0.63±0.01	**0.37**±0.00	**0.77**±0.00	**0.30**±0.00			
NHP	0.49±0.02	0.76±0.01	0.52±0.02	0.28±0.00	0.65±0.01	0.19±0.01			
A-NHP	0.47±0.03	0.73±0.04	0.48±0.04	0.28±0.00	0.64±0.03	0.18±0.01			
COTIC	0.43±0.02	0.69±0.03	0.43±0.03	0.28±0.00	0.61±0.02	0.16±0.01			

- On the Churn dataset, the AE yields the best results among generative models for global tasks, but it is outperformed by AR in local tasks. Contrastive approaches and TPP models are slightly behind in global quality but drastically underperform in the next MCC prediction.

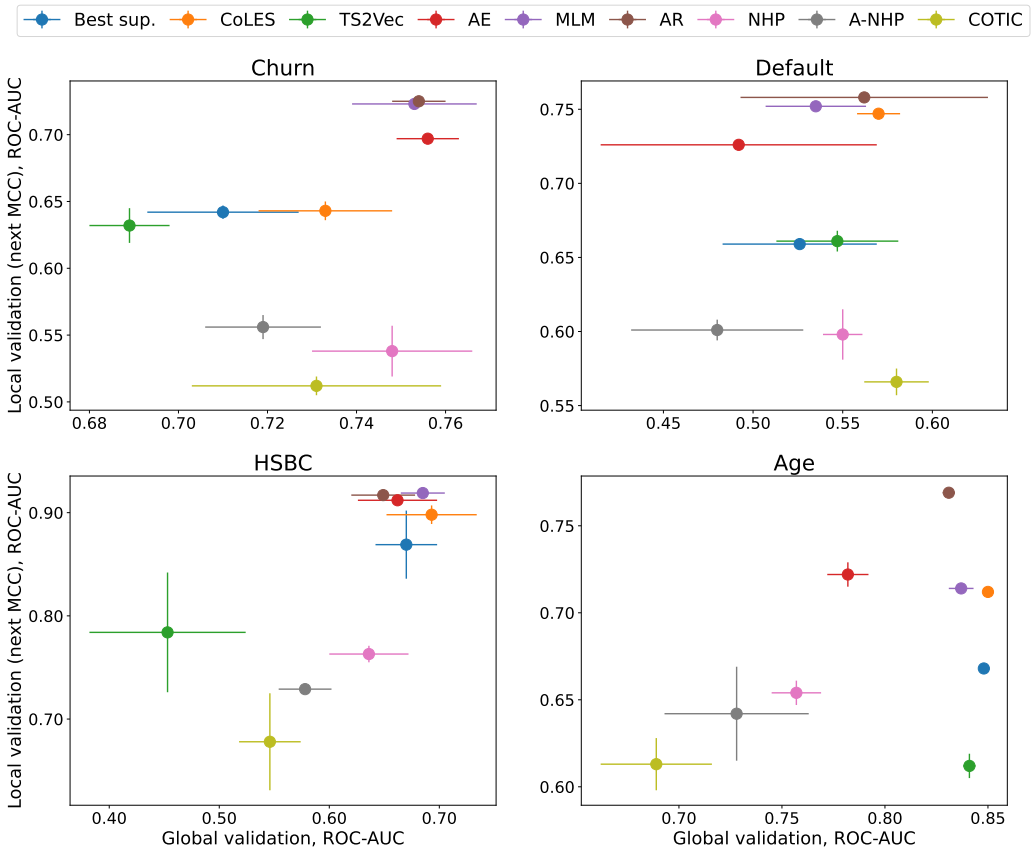

Figure 6: Quality of the models regarding their global and local properties. "Best sup." stands for the best supervised baseline. The x-axis at each graph corresponds to global validation ROC-AUC, while the y-axis shows the next MCC prediction ROC-AUC. Thus, the upper and righter the dot is, the better model it represents.

- For Default, the outcomes are more nuanced: the best global results are for CoLES and COTIC, while AR leads among generative approaches. For the next MCC prediction, the AR model is superior to others, followed by MLM.
- HSBC and Age results look similar to the Churn ones. However, CoLES and MLM are the best in terms of global properties. The local tasks are, again, dominated by the generative models (AR, MLM, and AE).
- In most cases, TPP models do not produce high-quality representations of event sequences. This issue can be explained as these methods

generally aim at another task — the intensity function restoration.

- Surprisingly, supervised models do not provide efficient embeddings in most cases. This could be due to the noise in the markup of the datasets we used.

Overall, the generative models (AR and MLM, in particular) look preferable regarding the trade-off between the local and the global properties of their embeddings. However, when selecting a specific solution, one must balance the models, considering the peculiarities of the downstream task at hand.

Results on the proprietary data. In Table 3, we provide the results of the experiments carried out on the industrial-scale private dataset described in Section 5. Note that standard deviation is absent: due to the size of this dataset, it is infeasible to collect statistics over several runs.

Table 3: Global and local validation ROC-AUC values on the private industrial-scale dataset. All metrics in the Table should be maximized. The best values are **highlighted with bold font**, and the second best values are underlined.

Method	Global validation: age	Global validation: gender	Local validation: next MCC
CoLES	**0.955**	**0.920**	0.750
TS2Vec	0.918	0.839	0.692
AE	0.886	0.753	0.734
MLM	0.939	0.896	0.734
AR	0.946	0.865	**0.770**
COTIC	0.919	0.840	0.684

The overall conclusion remains the same for this data: the generative model AR outperforms CoLES in the local task while falling slightly short of it on global properties. Another generative MLM approach also shows comparable results on both global and local problems.

6.2. Experiments on the Dynamics of Representations

We adhere to our methodology for investigating the dynamic properties of representations introduced in Section 3.3. The experiments are conducted for three models: CoLES, AE, and AR.

Dynamics assessment using artificial data. The experiment is designed as follows. We consider 100 pairs of clients. For each pair, we substitute the second half of one client's transactions with another's. As a result, the first client's history now contains an artificial change point. Then, we measure the distances between the clients' local representations: we assume they are significant before the change point and start to decrease right after it.

We also perform the opposite experiment (by replacing the first half of the transactions instead of the second): initially, the synthesized clients' sequences are similar, but after the change point, their behavior diverges. The distances are expected to increase accordingly.

The resulting dynamic for setups described above is depicted in Figure 7. Both AE and AR react to changes faster than CoLES. The figure also demonstrates a difference in convergence speed: for generative models, it appears to be exponentially fast, whereas for the contrastive one, it is closer to linear.

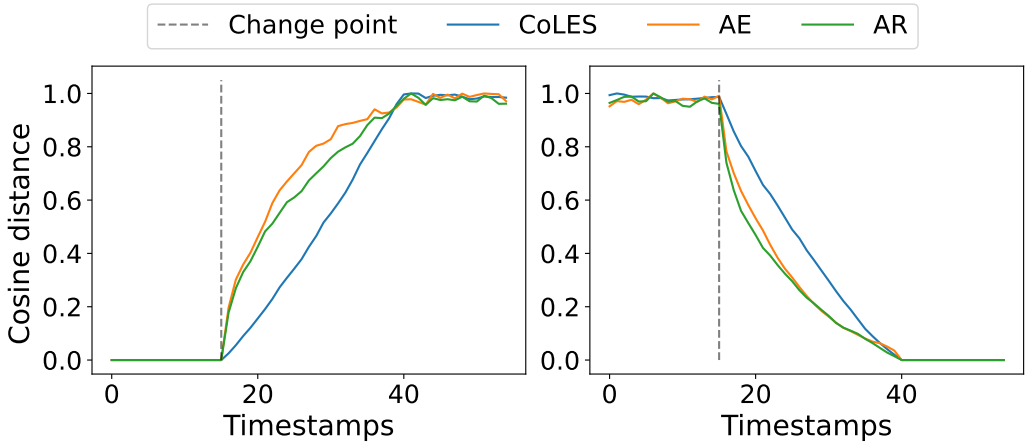

Figure 7: Distances between sequences' local representations. Left figure: clients' behavior diverges; right figure: it converges. Results are averaged over 100 sequences.

Dynamics of real-world data embeddings. In this experiment, we use a small subset of the Churn dataset consisting of 35 clients whose transaction history contains intrinsic change points. As such change moments, we consider changes in the transaction currency.

We build local representations of the above data via pre-trained models and run a Change Point Detection (CPD) procedure on top of them. The

detection accuracy and detection delay (two important metrics in the CPD domain) are shown in Figure 8.

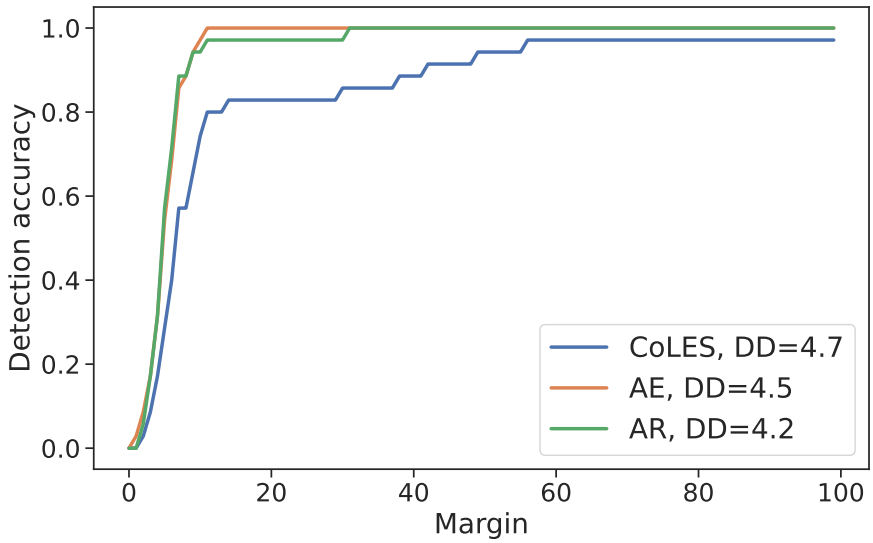

Figure 8: Dependency of CPD accuracy on the margin m. Mean detection delays for each model are provided in the legend. We want to maximize detection accuracy and minimize detection delay.

AE and AR provide higher detection accuracy than CoLES for small margins. Moreover, the use of AR embeddings results in the lowest detection delays. It can be concluded that generative models are more sensitive to local pattern changes than contrastive ones.

6.3. Result on improving embeddings via external information

In this section, we describe the results of experiments aimed at obtaining an external context representation and using it to improve existing models.

The context vector is built on embeddings from a pre-trained CoLES encoder. To this end, we investigate various types of representation aggregations: averaging (*Mean*), maximization (*Max*), and an attention mechanism, with or without a trainable matrix (*Learnable attention* and *Attention*). The results are compared with a conventional CoLES encoder without adding external information (*No pooling*).

As in previous experiments, we demonstrate our results in two ways. Quantitative metrics are presented in Table 4, while qualitative trade-offs

between global and local tasks are shown in Figure 9. All results are averaged across five different pre-trained encoder models.

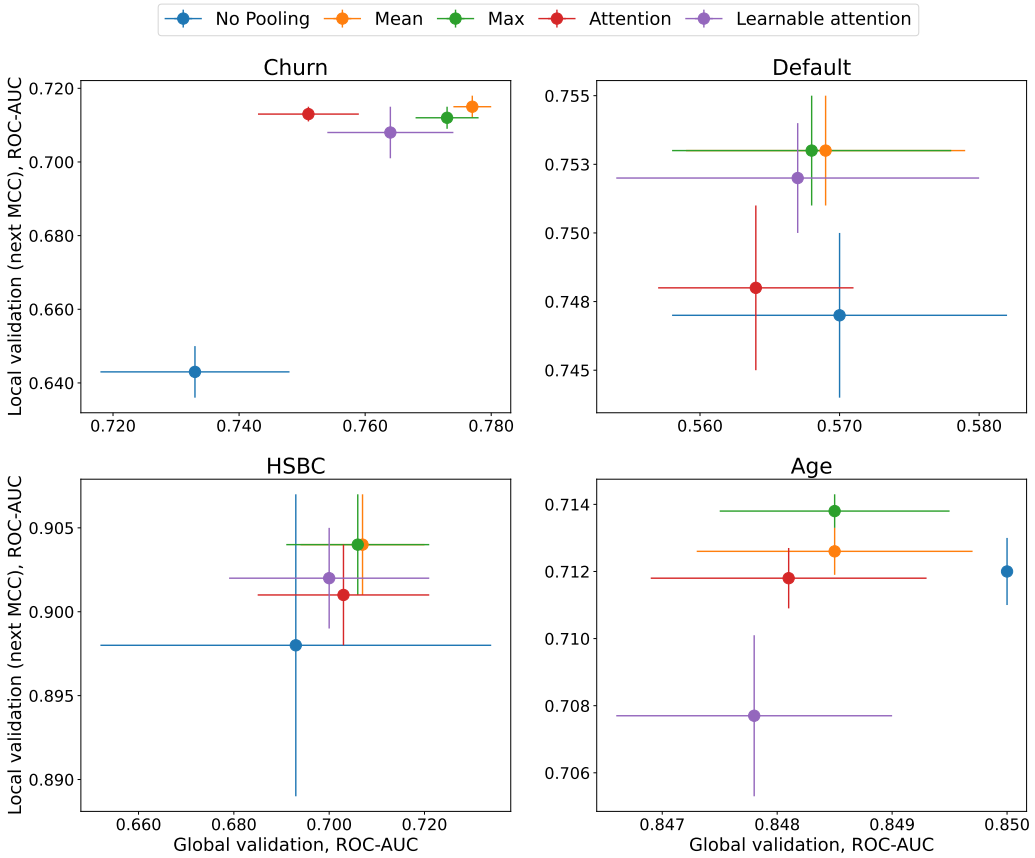

Figure 9: Quality of the various types of aggregation regarding their global and local properties. The x-axis at each graph corresponds to global validation ROC-AUC, while the y-axis shows the next MCC prediction ROC-AUC. Thus, the upper and righter the dot is, the better model it represents.

The experimental results indicate that using external context generally leads to improved performance. This is especially evident in the case of the balanced Churn dataset, where improvements can be noticed in all testing scenarios.

The Attention and Learnable attention approaches are often among the most effective, or at least comparable to the best approaches, especially in local validation tasks. It meets our expectations as the attention mechanism is known to assist in identifying local patterns within sequences (see 4.5).

Table 4: Quality metrics for global and local embedding validation results for external information addition. All metrics in the Table should be maximized. The results are averaged by five runs and are given in the format *mean* ± *std*. The best values are **highlighted**.

Method	Global validation			Local validation: next MCC			Local validation: binary target		
	Accuracy	ROC-AUC	PR-AUC	Accuracy	ROC-AUC	PR-AUC	Accuracy	ROC-AUC	PR-AUC
Churn									
No pooling	0.67±0.02	0.73±0.02	0.77±0.03	0.23±0.00	0.64±0.01	0.16±0.01	**0.73**±0.00	0.56±0.01	0.32±0.01
Mean	**0.71**±0.01	**0.78**±0.00	**0.83**±0.00	0.26±0.01	**0.72**±0.00	**0.21**±0.00	**0.73**±0.00	0.59±0.01	0.35±0.01
Max	**0.71**±0.01	0.77±0.01	0.82±0.01	0.26±0.01	0.71±0.00	**0.21**±0.00	**0.73**±0.00	0.65±0.03	0.40±0.02
Attention	0.69±0.01	0.75±0.01	0.80±0.01	**0.27**±0.01	0.71±0.00	**0.21**±0.00	**0.73**±0.00	0.60±0.00	0.34±0.01
Learn. attention	0.70±0.01	0.76±0.01	0.82±0.00	0.26±0.01	0.71±0.01	**0.21**±0.00	0.71±0.00	**0.67**±0.00	**0.43**±0.01
Default									
No pooling	0.96±0.00	**0.57**±0.01	0.06±0.01	0.34±0.01	0.75±0.00	0.26± 0.00	0.99±0.00	0.54±0.04	0.01±0.00
Mean	0.96±0.00	**0.57**±0.01	0.06±0.01	0.34±0.01	0.75±0.00	**0.27**±0.00	0.99±0.00	**0.65**±0.06	0.01±0.00
Max	0.96±0.00	**0.57**±0.01	0.06±0.01	0.34±0.01	0.75±0.00	**0.27**±0.00	0.99±0.00	0.44±0.11	0.01±0.01
Attention	0.96±0.00	0.56±0.01	0.06±0.01	0.34±0.01	0.75±0.00	0.26±0.00	0.99±0.00	0.61±0.06	0.01±0.00
Learn. attention	0.96±0.00	**0.57**±0.01	0.06±0.01	0.34±0.01	0.75±0.00	**0.27**±0.00	0.99±0.00	0.64±0.09	0.01±0.00
HSBC									
No pooling	0.92±0.00	0.69±0.04	0.15±0.02	0.69±0.02	0.90±0.01	0.82±0.01	1.00±0.00	0.38±0.09	0.01±0.00
Mean	0.92±0.00	**0.71**±0.01	**0.25**±0.01	0.69±0.02	0.90±0.00	**0.83**±0.01	1.00±0.00	0.39±0.21	0.01±0.01
Max	0.92±0.00	**0.71**±0.02	0.21±0.02	0.70±0.02	0.90±0.00	**0.83**±0.01	1.00±0.00	0.41±0.19	0.01±0.00
Attention	0.92±0.00	0.70±0.02	0.18±0.01	**0.74**±0.02	0.90±0.00	**0.83**±0.01	1.00±0.00	**0.47**±0.20	0.01±0.00
Learn. attention	0.92±0.00	0.70±0.02	0.22±0.02	0.70±0.02	0.90±0.00	**0.83**±0.01	1.00±0.00	0.38±0.14	0.01±0.00
Age									
No pooling	0.61±0.00	0.85±0.00	**0.66**±0.00	0.32±0.00	0.71±0.00	0.24±0.00			
Mean	**0.64**±0.03	0.85±0.00	0.63±0.02	**0.33**±0.00	0.71±0.00	0.24±0.00			
Max	**0.64**±0.03	0.85±0.00	0.63±0.02	**0.33**±0.00	0.71±0.00	0.24±0.00		NA	
Attention	**0.64**±0.03	0.85±0.00	0.63±0.02	**0.33**±0.00	0.71±0.00	0.24±0.00			
Learn. attention	0.61±0.01	0.85±0.00	0.65±0.00	**0.33**±0.00	0.71±0.00	0.24±0.00			

The Mean and Max methods are often the top-1 and top-2 performers in global validation tasks, respectively. This behavior is also explainable, as both approaches smooth local patterns while aggregating representations. While this may be beneficial for global properties, it negatively affects the quality of local ones.

6.4. Ranking and Discussion

In this research, we compare several classes of transactional data models in terms of their embeddings' global and local properties. This section attempts to summarize these findings by in-depth analysis and supplementary ranking with respect to ROC-AUC values for each task.

The rankings compare all the models from Table 2, as well as "CoLES ext." – the CoLES model, modified by the mean external information inclusion procedure, which is a reasonable compromise judging by Table 4. This is more illustrative than the specific metric values from Table 2 since here we equate the statistically indistinguishable results. We increase the rank if

the p-value is less than 0.1 for a one-sided T-test, i.e. only if the increment is large enough compared to the corresponding variances.

6.4.1. Event type.

The resulting per-dataset ranks for the local event type task are presented in Table 5. As stated above, the autoregressive model is a clear winner here: this indicates a robustly superior ability to capture local patterns and use them for forecasting.

The MLM approach makes a close second, dropping to third place only on the Age dataset, where it also gives in to the AE model. This can be explained by a greater dependence on the order in which transactions are made: the Transformer model used in MLM is known for ignoring positional information (see e.g. [70]), while the RNN used in AE takes it into account.

Next, interestingly enough, CoLES with external information slightly outperforms the AE model on average. It seems that the external information gives the edge necessary for the advanced CoLES method to sometimes beat a simple but specialized autoencoder on local tasks, providing statistics on how other clients differ from each other at a specific moment in time. Overall, as we stated previously, generative methods outperform vanilla contrastive methods, with the latter performing on-par with supervised baselines.

The ranking concludes with the temporal point process baselines. This agrees with prior research: authors of [71] state that TPP methods require long sequences to operate successfully, which is not the case for the local task.

6.4.2. Local binary.

The ranks for the local binary task are presented in Table 6. Overall, contrary to the previously considered event-type prediction task, the generative methods now lag behind the modified CoLES method regarding the mean model rank. Specifically, the generative methods show poor performance on the Default dataset. This can be explained by the label's more global nature: after all, it was deduced from a sequence-wise, global label via cropping, and the probability of loan repayment likely changes slowly with time. Moreover, the highly-local AR model comes in fourth here, which further supports this claim. Another very important factor here is class imbalance. Remember, the Default dataset is highly imbalanced by itself; our local label extraction procedure only makes matters worse. It can be concluded, that contrastive

Table 5: Model ranking for the local event type prediction task. Equal ranks correspond to indistinguishable performance. Models are colour-coded with superscripts for monochrome versions: blue[§] for generative, green[†] for contrastive and fuchsia[‡] for TPP.

	Age	Churn	Default	HSBC	Mean
AR[§]	1	1	1	1	1.00
MLM[§]	3	1	2	1	1.75
CoLES ext.[†]	3	2	2	3	2.50
AE[§]	2	3	4	2	2.75
CoLES[†]	3	4	3	3	3.25
Best baseline	4	4	5	3	4.00
TS2Vec[†]	6	4	5	4	4.75
A-NHP[‡]	5	5	6	4	5.00
NHP[‡]	5	5	6	4	5.00
COTIC[‡]	6	6	7	5	6.00

and TPP models handle such extreme class asymmetry better, able to extract rare patterns, while generative models focus mostly on statistically likely ones.

The Churn dataset does not distinguish much between generative methods and "CoLES ext.": the label here seems to require both local and global properties. Lastly, the HSBC dataset, which initially had a local label (indicating specifically which transactions are fraudulent), favours generative models, which is in line with our reasoning.

Finally, it can be observed that COTIC and TS2Vec perform surprisingly well on HSBC. Both of these models use convolutions. Good performance may be due to the fact that they are able to meticulously analyse short, fixed-length sequences, paying attention (via max pooling) to the most suspicious transaction.

6.4.3. Global.

The ranks for the global task are presented in Table 7. It comes without surprise that the CoLES variations take the lead here, with the external information model showing more stable performance than the vanilla option. Specifically, the plain CoLES lags behind on the Churn task. The final dynamics right before a client leaves the bank are presumably very important here, making Churn somewhat local, and, as mentioned previously, adding external information improves CoLES's performance on local tasks. This

Table 6: Model ranking for the local binary label prediction task. Equal ranks correspond to indistinguishable performance. Models are colour-coded with superscripts for monochrome versions: blue[§] for generative, green[†] for contrastive and fuchsia[‡] for TPP.

	Churn	Default	HSBC	Mean
CoLES ext.[†]	1	1	3	1.67
MLM[§]	2	2	1	1.67
AR[§]	1	4	1	2.00
AE[§]	2	2	3	2.33
TS2Vec[†]	4	1	2	2.33
A-NHP[‡]	3	1	4	2.67
NHP[‡]	3	2	3	2.67
CoLES[†]	3	2	4	3.00
Best baseline	3	3	4	3.33
COTIC[‡]	4	5	2	3.67

point is supported by the fact that the contrastive TS2Vec method also performs poorly on the Churn task, while the local generative methods show good results here. A more interesting observation is that TPP methods rise up to a respectable second place on Churn: a client's transactions grow less frequent before he leaves the bank, which makes this task very well suited for these time-sensitive models.

Next come the generative models, which mostly show good results here. A notable exception is provided by our best event type model, AR: it loses to MLM, AE, and even the supervised baseline for the HSBC task. It seems that its autoregressive nature renders it too local. It has a very narrow view of only the few last transactions, making it forget fraudulent ones that may have appeared early on in the sequence.

The ranking concludes with TPP methods, showing highly unstable performance between datasets. All these models take into the account the temporal structure of the sequence, which appears to be a dead weight for the global task, only hindering their learning.

This table presents several anomalies. For one, the supervised baseline is ranked first for the Age task. As mentioned above, the best way to handle Age is to look for certain locally ordered patterns within the sequence, which is exactly what a simple RNN does. Since the sequences are long, their lengths vary little (in a narrow range from 700 to a little over 1k elements) and there is plenty of labelled data, the baseline is able to accumulate enough

information to tune a good, specialized classifier.

Next, the NHP model achieves high performance on the Churn task. As we have concluded from the local binary label, churning may be predictable based on the time intervals closer to the end of the sequence. NHP appears to capture these dynamics well, without introducing restrictive complex modifications.

Finally, COTIC performs on par with the external information CoLES on Default. This supports our hypothesis that the continuous-time convolutions work well on highly regular sequence sizes (all sequences from default are of length 300), extracting local and global patterns from fixed positions in the sequence and combining them into highly informative representations.

Table 7: Model ranking for the global label prediction task. Equal ranks correspond to indistinguishable performance. Models are colour-coded with superscripts for monochrome versions: blue[§] for generative, green[†] for contrastive and fuchsia[‡] for TPP.

	Age	Churn	Default	HSBC	Mean
CoLES ext.[†]	1	1	1	1	1.00
CoLES[†]	1	3	1	1	1.50
MLM[§]	2	2	2	2	2.00
Best baseline	1	4	2	2	2.25
AR[§]	3	2	1	3	2.25
AE[§]	4	2	2	2	2.50
NHP[‡]	5	2	2	3	3.00
COTIC[‡]	6	3	1	4	3.50
TS2Vec[†]	2	5	2	5	3.50
A-NHP[‡]	5	3	3	4	3.75

6.4.4. Executive Summary

Now, we will briefly summarize the above discussion in a few points. Overall, the generative methods clearly lead at the event-type task (since they were effectively trained for it). The local binary and global tasks offer fine-grained insights, based on the label's nature.

- The Age dataset has regular, simple, ordered patterns, appearing all throughout the sequence, indicative of the final label. The event-type task is thus best solved by ordered generative models (using RNN as the backbone), and the global one, having enough transactions and data – by the supervised baseline.

- The Churn label is largely defined by the time intervals closer to the end of the sequence. Consequently, this task requires a compromise between local and global properties, best achieved by CoLES with the inclusion of extrnal information. The local binary task, although devised from a global label, also relies greatly on fine-grained dynamics. The AR performs well here, on par with "CoLES ext.".

- The HSBC dataset has a truly local binary label, which makes it well-suited for generative models. However, the continuous-convolution approaches are also a great fit for this task: the cropped sequence length is fixed during testing, so they are able to pay attention to the most suspicious transactions via max-pooling. On the other hand, the global task requires broad attention to whole sequences of different lengths, favouring contrastive approaches over generative and TPP methods.

- The Default dataset has proven to be the hardest for all the considered models, without any clear winners on downstream tasks. The local binary label seems to favour contrastive and TPP models: we argue this is due to the label leaning more on global properties, as well as extreme class imbalance. Unsurprisingly, both CoLES methods take first place in the global ranking. However, the victory is shared with continuous convolution methods: the sequences here are also of fixed size, both during testing and training, which appears to be the speciality of these approaches.

7. Conclusions

This paper examines various methods for extracting meaningful representations from transaction data, evaluating their effectiveness in capturing both global and local patterns of transactional sequences. We adapted generative self-supervised learning methodologies (specifically AE, MLM, and AR models) to the domain of transaction sequences, resulting in embeddings that effectively capture local and dynamic features, while reasonably addressing global patterns inherent in transactional data. These models were rigorously compared against state-of-the-art domain-specific deep learning techniques, including contrastive self-supervised learning (such as CoLES and TS2Vec), temporal point process models (HNP, A-NHP, and COTIC), and conventional supervised learning approaches.

Our comprehensive benchmark, i.e. one that simultaneously addresses all desired representation properties, reveals that no single method is universally superior for all applied problems. Contrastive approaches generally excel in global classification tasks, making them effective for situations requiring a broader view of transactional data. In contrast, generative models are better suited for capturing local patterns, essential for scenarios involving immediate customer interactions and behavior prediction. Temporal point process models, which are typically designed for process intensity restoration, tend to underperform in representation learning contexts. However, their ability to account for time intervals makes them a great choice for certain scenarios. Finally, the convolution models (TS2Vec, COTIC) excel at uniform-length sequences.

Among the classic methods evaluated, the proposed AR model stands out for its robust performance across various tasks. It slightly gives way to CoLES, yet it remains comparably effective in global classification while surpassing all other models in next MCC code prediction tasks, indicating its ability to effectively balance the recognition of both global and local patterns within transactional data.

Additionally, we introduced an innovative method for incorporating external contextual information into transactional representations. By leveraging the activity of other clients, this method constructs an external context vector, significantly improving model performance, with an increase in ROC-AUC of up to 20% for local tasks, making it the clear winner in our downstream benchmarks. The best results were achieved using a trainable attention mechanism that accounts for embeddings of similar clients, however the simpler mean aggregation also gives a comparable boost in metrics, making it a reasonable compromise.

Overall, our research provides a benchmark for evaluating diverse properties of various representation learning methods applied to financial transaction data. These techniques offer scalable and adaptable solutions that can be applied across a wide range of banking applications, enhancing decision-making and customer experience. The contributions made to representation learning in this paper have significant potential for advancing the banking industry. By effectively utilizing AI, banks can optimize operations, reduce risks, and offer more personalized services to their clients. As AI continues to develop, these findings and methods will drive further innovation in financial services.

8. Acknowledgments

The research was supported by the Russian Science Foundation grant 20-7110135.

Appendix A. Implementation details

Main models. In this research, we generally follow the pipeline of the pytorch-lifestream Python package, which contains the implementation of the CoLES model. Additionally, we use NHP and A-NHP models from the EasyTPP [72] library available online. Finally, the COTIC model implementation was taken from the original GitHub repository and adopted into our pipeline.

Information about the architectures and hyperparameters of the models used in our study is given below.

- CoLES is a one-layer recurrent neural network. We use LSTM with a hidden layer dimension of 1024 for the Churn, Age, and HSBC datasets and 512 for industrial data. For Default, we select GRU with a hidden size of 800. The models were trained for 60 epochs with a batch size of 128.

- AE's encoder and AR have identical architectures for all the public datasets — LSTM with a hidden layer dimension of 1024. The hidden size of 512 was used for industrial data. AR was trained for 60 epochs with a batch size of 128.

- The AE decoder is an LSTM with a hidden layer dimension twice bigger than in the encoder, 1024 for industrial data and 2048 for other datasets. The model was trained for 2000 steps with a batch size of 1024.

- The MLM is a Transformer with six hidden layers with eight heads; the embedding dimension and the perceptron hidden layer dimension equals 512 for industrial data and 1024 for other datasets. The training regimen is similar to that of the AE's decoder: 2000 steps on batches of 1024.

- For AR, MLM, and AE models, we use a combination of two losses described in 4.3. The weights for these losses are set to five and one, respectively.

- TS2Vec has ten one-dimensional convolutional blocks with an expanded kernel size of three, an expansion factor of 2^l (where l is the block number), and the number of output channels equals 512 for private data and 1024 for other datasets. It was trained for 100 epochs with batches of 1024.

- The COTIC model has five continuous one-dimensional convolutional layers with a hidden dimension of 32 and a kernel size of five. Its kernel is a multi-layer perceptron with hidden layer dimensions eight, four, and eight, and the prediction head is a two-layer perceptron with a hidden dimension of 100. The model was trained for 100 epochs with a batch size of 20. We had to significantly limit the number of parameters to reduce the memory consumption and the training time since COTIC is inherently less efficient than other models.

- NHP has a custom continuous-time LSTM architecture with exponential decay of hidden states as in original work [43]. We mainly adopt the parameters from the EasyTPP experiment configuration files while increasing the hidden dimension of the model up to 64 to be comparable with the other TPP models. It was trained for 50 epochs with a batch size of 128.

- The encoder of the A-NHP model consists of two multi-head attention blocks with a final embedding dimension of 64. It also uses a temporal encoding with a time embedding size of four. The model was trained for 50 epochs with batches of 32.

Input features. Note that all self-supervised approaches (both contrastive and generative) take two features as input: MCCs and transaction amounts. As MCC is a categorical variable, we use its embedding of size 24 (for Churn and HSBC) or 16 (for Age and Default).

In contrast, the considered TPP models (NHP, A-NHP, COTIC) do not process amounts but take times between events as a part of the input. Here, we cannot use raw UNIX timestamps due to numeric overflows. Thus, all the times are normalized to represent the number of days (float) since the first transaction in the dataset. Furthermore, to foster the quality of next event type prediction, we clip the rare MCC codes as it is done for the autoencoder models 4.3.

Models with external context. A model using external information requires quite a lot of computational resources since it needs to store local representations of all users from the training set for all available time points (Subsection 4.5). Due to memory constraints, we store only a random subset of local representations: for the Churn dataset, the number of clients to train in all experiments was 500; for the Default dataset — 150; for HSBC — 300; and for Age — 2000. Additionally, the batch is reduced to eight. Other hyperparameters for the encoder remain the same as in the experiments with the regular CoLES model.

Validation model parameters. For the local validation procedure outlined in Section 3, we used the following hyperparameter values: window size is 32, shift step is 16, batch size is 512, and the maximum number of epochs is ten.

References

[1] A. Bany Mohammed, M. Al-Okaily, D. Qasim, M. Khalaf Al-Majali, Towards an understanding of business intelligence and analytics usage: Evidence from the banking industry, International Journal of Information Management Data Insights 4 (1) (2024) 100215. doi:https://doi.org/10.1016/j.jjimei.2024.100215.

[2] L. A. Bueno, T. F. Sigahi, I. S. Rampasso, W. Leal Filho, R. Anholon, Impacts of digitization on operational efficiency in the banking sector: Thematic analysis and research agenda proposal, International Journal of Information Management Data Insights 4 (1) (2024) 100230. doi:https://doi.org/10.1016/j.jjimei.2024.100230.

[3] A. Amato, J. R. Osterrieder, M. R. Machado, How can artificial intelligence help customer intelligence for credit portfolio management? a systematic literature review, International Journal of Information Management Data Insights 4 (2) (2024) 100234. doi:https://doi.org/10.1016/j.jjimei.2024.100234.

[4] O. Kaya, J. Schildbach, D. B. AG, S. Schneider, Artificial intelligence in banking, Artificial intelligence (2019).

[5] V. Singh, S.-S. Chen, M. Singhania, B. Nanavati, A. kumar kar, A. Gupta, How are reinforcement learning and deep learning algorithms used for big data based decision making in financial industries–a review and research agenda, International Journal of Information Management Data Insights 2 (2) (2022) 100094. doi:https://doi.org/10.1016/j.jjimei.2022.100094.

[6] J. Jurgovsky, M. Granitzer, K. Ziegler, S. Calabretto, P.-E. Portier, L. He-Guelton, O. Caelen, Sequence classification for credit-card fraud detection, Expert Systems with Applications 100 (2018) 234–245. doi:10.1016/j.eswa.2018.01.037.

[7] D. Babaev, N. Ovsov, I. Kireev, M. Ivanova, G. Gusev, I. Nazarov, A. Tuzhilin, CoLES: Contrastive learning for event sequences with self-supervision, in: Proceedings of the 2022 International Conference on Management of Data, ACM, New York, NY, USA, 2022. doi:10.1145/3514221.3526129.

[8] R. Bin Sulaiman, V. Schetinin, P. Sant, Review of machine learning approach on credit card fraud detection, Human-Centric Intelligent Systems 2 (1-2) (2022) 55–68. doi:10.1007/s44230-022-00004-0.

[9] T. Li, G. Kou, Y. Peng, A new representation learning approach for credit data analysis, Information Sciences 627 (2023) 115–131. doi:10.1016/j.ins.2023.01.068.

[10] R. A. Mancisidor, M. Kampffmeyer, K. Aas, R. Jenssen, Learning latent representations of bank customers with the variational autoencoder, Expert Systems with Applications 164 (2021) 114020. doi:10.1016/j.eswa.2020.114020.

[11] E. Romanenkova, A. Rogulina, A. Shakirov, N. Stulov, A. Zaytsev, L. Ismailova, D. Kovalev, K. Katterbauer, A. AlShehri, Similarity learning for wells based on logging data, Journal of Petroleum Science and Engineering 215 (110690) (Aug. 2022). doi:10.1016/j.petrol.2022.110690.

[12] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, F. Makedon, A survey on contrastive self-supervised learning, Technologies (Basel) 9 (1) (2020) 2. doi:10.3390/technologies9010002.

[13] V. Zhuzhel, V. Grabar, G. Boeva, A. Zabolotnyi, A. Stepikin, V. Zholobov, M. Ivanova, M. Orlov, I. Kireev, E. Burnaev, R. Rivera-Castro, A. Zaytsev, Continuous-time convolutions model of event sequences, arXiv preprint arXiv:2302.06247 (2023). doi:10.48550/arXiv.2302.06247.

[14] Y. Heryadi, H. L. H. S. Warnars, Learning temporal representation of transaction amount for fraudulent transaction recognition using CNN, Stacked LSTM, and CNN-LSTM, in: 2017 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom), IEEE, Phuket, 2017, pp. 84–89. doi:10.1109/CYBERNETICSCOM.2017.8311689.

[15] M. Ala'raj, M. F. Abbod, M. Majdalawieh, L. Jum'a, A deep learning model for behavioural credit scoring in banks, Neural Computing and Applications 34 (8) (2022) 5839–5866. doi:10.1007/s00521-021-06695-z.

[16] S. Deldari, D. V. Smith, H. Xue, F. D. Salim, Time series change point detection with self-supervised contrastive predictive coding, in: Proceedings of the Web Conference 2021, ACM, New York, NY, USA, 2021. doi:10.1145/3442381.3449903.

[17] M. Begicheva, A. Zaytsev, Bank transactions embeddings help to uncover current macroeconomics, in: 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2021. doi:10.1109/ICMLA52953.2021.00276.

[18] L. C. Thomas, A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers, International journal of forecasting 16 (2) (2000) 149–172. doi:10.1016/S0169-2070(00)00034-0.

[19] M. Gomez-Rodriguez, D. Balduzzi, B. Schölkopf, Uncovering the temporal dynamics of diffusion networks, in: International Conference on Machine Learning, 2011.
URL https://api.semanticscholar.org/CorpusID:12901792

[20] X. Dastile, T. Celik, M. Potsane, Statistical and machine learning models in credit scoring: A systematic literature survey, Applied Soft Computing 91 (2020) 106263. doi:10.1016/j.asoc.2020.106263.

[21] B. R. Gunnarsson, S. Vanden Broucke, B. Baesens, M. Óskarsdóttir, W. Lemahieu, Deep learning for credit scoring: Do or don't?, European Journal of Operational Research 295 (1) (2021) 292–305. doi:10.1016/j.ejor.2021.03.006.

[22] L. Marceau, L. Qiu, N. Vandewiele, E. Charton, A comparison of deep learning performances with other machine learning algorithms on credit scoring unbalanced data, arXiv preprint arXiv:1907.12363 (2019). doi:10.48550/arXiv.1907.12363.

[23] V. Moscato, A. Picariello, G. Sperlí, A benchmark of machine learning approaches for credit score prediction, Expert Systems with Applications 165 (2021) 113986. doi:10.1016/j.eswa.2020.113986.

[24] M. Schmitt, Deep learning vs. gradient boosting: Benchmarking state-of-the-art machine learning algorithms for credit scoring, arXiv preprint arXiv:2205.10535 (2022). doi:10.48550/arXiv.2205.10535.

[25] A. C. Bahnsen, D. Aouada, A. Stojanovic, B. Ottersten, Feature engineering strategies for credit card fraud detection, Expert Systems with Applications 51 (2016) 134–142. doi:10.1016/j.eswa.2015.12.030.

[26] A. Zaytsev, A. Natekin, E. Vorsin, V. Smirnov, O. Sidorshin, A. Senin, A. Dudin, D. Berestnev, Designing an attack-defense game: how to increase robustness of financial transaction models via a competition, arXiv preprint arXiv:2308.11406 (2023).

[27] X. Zhang, Y. Han, W. Xu, Q. Wang, HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture, Information Sciences 557 (2021) 302–316. doi:10.1016/j.ins.2019.05.023.

[28] D. Babaev, M. Savchenko, A. Tuzhilin, D. Umerenkov, ET-RNN: Applying deep learning to credit loan applications, in: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, ACM, New York, NY, USA, 2019, pp. 2183–2190. doi:10.1145/3292500.3330693.

[29] C. Wang, D. Han, Q. Liu, S. Luo, A deep learning approach for credit scoring of peer-to-peer lending using attention mechanism LSTM, IEEE Access 7 (2018) 2161–2168. doi:10.1109/ACCESS.2018.2887138.

[30] C. Wang, Z. Xiao, A deep learning approach for credit scoring using feature embedded transformer, Applied Sciences (Basel) 12 (21) (2022) 10995. doi:10.3390/app122110995.

[31] Y. Xie, G. Liu, C. Yan, C. Jiang, M. Zhou, M. Li, Learning Transactional Behavioral Representations for Credit Card Fraud Detection, IEEE Transactions on Neural Networks and Learning Systems PP (2022) 1–14, conference Name: IEEE Transactions on Neural Networks and Learning Systems. doi:10.1109/TNNLS.2022.3208967.

[32] J. Forough, S. Momtazi, Ensemble of deep sequential models for credit card fraud detection, Applied Soft Computing 99 (2021) 106883. doi:10.1016/j.asoc.2020.106883.

[33] Q. Zhang, A. Lipani, O. Kirnap, E. Yilmaz, Self-attentive Hawkes process, in: H. D. III, A. Singh (Eds.), Proceedings of the 37th In-

ternational Conference on Machine Learning, Vol. 119 of Proceedings of Machine Learning Research, PMLR, 2020, pp. 11183–11193. doi:10.48550/arXiv.1907.07561.

[34] Z. Li, G. Liu, C. Jiang, Deep Representation Learning With Full Center Loss for Credit Card Fraud Detection, IEEE Transactions on Computational Social Systems 7 (2) (2020) 569–579. doi:10.1109/TCSS.2020.2970805.

[35] V. Moskvoretskii, D. Osin, E. Shvetsov, I. Udovichenko, M. Zhelnin, A. Dukhovny, A. Zhimerikina, A. Efimov, E. Burnaev, Self-supervised learning in event sequences: A comparative study and hybrid approach of generative modeling and contrastive learning, arXiv preprint arXiv:2401.15935 (2024).

[36] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, B. Xu, TS2Vec: Towards universal representation of time series, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, Association for the Advancement of Artificial Intelligence (AAAI), 2022, pp. 8980–8987. doi:10.1609/aaai.v36i8.20881.

[37] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.

[38] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, OpenAI blog 1 (8) (2019) 9.

[39] M. Tschannen, O. Bachem, M. Lucic, Recent advances in autoencoder-based representation learning, arXiv preprint arXiv:1812.05069 (2018). doi:10.48550/arXiv.1812.05069.

[40] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, R. Girshick, Masked autoencoders are scalable vision learners, in: 2022 IEEE/CVF Conference on

Computer Vision and Pattern Recognition (CVPR), IEEE, 2022, pp. 16000–16009. doi:10.1109/CVPR52688.2022.01553.

[41] A. G. Hawkes, Spectra of some self-exciting and mutually exciting point processes, Biometrika 58 (1) (1971) 83–90. doi:10.1093/biomet/58.1.83.

[42] T. Liniger, Multivariate Hawkes processes, Ph.D. thesis, ETH Zurich (2009).

[43] H. Mei, J. Eisner, The neural Hawkes process: A neurally self-modulating multivariate point process, in: Advances in neural information processing systems, Vol. 30, Long Beach, 2017. doi:10.48550/arXiv.1612.09328.

[44] H. Mei, C. Yang, J. Eisner, Transformer embeddings of irregularly spaced events and their participants, in: Proceedings of the 10th International Conference on Learning Representations (ICLR), 2022. doi:10.48550/arXiv.2201.00044.

[45] S. Zuo, H. Jiang, Z. Li, T. Zhao, H. Zha, Transformer Hawkes process, in: H. D. III, A. Singh (Eds.), Proceedings of the 37th International Conference on Machine Learning, Vol. 119 of Proceedings of Machine Learning Research, PMLR, 2020, pp. 11692–11702. doi:10.48550/arXiv.2002.09291.

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017). doi:10.48550/arXiv.1706.03762.

[47] S. C.-X. Li, B. Marlin, Learning from irregularly-sampled time series: A missing data perspective, in: Proceedings of the 37th International Conference on Machine Learning, PMLR, 2020, pp. 5937–5946. doi:10.48550/arXiv.2008.07599.

[48] H. Shi, Y. Zhang, H. Wu, S. Chang, K. Qian, M. Hasegawa-Johnson, J. Zhao, Continuous CNN for nonuniform time series, in: ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, IEEE, 2021, pp. 3550–3554. doi:10.1109/ICASSP39728.2021.9414318.

[49] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, LightGBM: A highly efficient gradient boosting decision tree, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in neural information processing systems, Vol. 30, Curran Associates, Inc., 2017, pp. 3146–3154.
URL https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[50] G. J. Van den Burg, C. K. Williams, An evaluation of change point detection algorithms, arXiv preprint arXiv:2003.06222 (2020). doi:10.48550/arXiv.2003.06222.

[51] E. Romanenkova, A. Stepikin, M. Morozov, A. Zaytsev, InDiD: Instant disorder detection via a principled neural network, in: Proceedings of the 30th ACM International Conference on Multimedia, ACM, New York, NY, USA, 2022. doi:10.1145/3503161.3548182.

[52] C. Truong, L. Oudre, N. Vayatis, Selective review of offline change point detection methods, Signal Processing 167 (2020) 107299. doi:10.1016/j.sigpro.2019.107299.

[53] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Stroudsburg, PA, USA, 2014, p. 1724–1734. doi:10.3115/v1/D14-1179.

[54] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, J. Tang, Self-supervised learning: Generative or contrastive, IEEE Transactions on Knowledge and Data Engineering 35 (1) (2023) 857–876. doi:10.1109/TKDE.2021.3090866.

[55] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[56] R. Hadsell, S. Chopra, Y. LeCun, Dimensionality reduction by learning an invariant mapping, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), Vol. 2, 2006, pp. 1735–1742. doi:10.1109/CVPR.2006.100.

[57] R. R. Chowdhury, J. Li, X. Zhang, D. Hong, R. K. Gupta, J. Shang, PrimeNet: Pre-training for irregular multivariate time series, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 37, Association for the Advancement of Artificial Intelligence (AAAI), 2023, pp. 7184–7192. doi:10.1609/aaai.v37i6.25876.

[58] I. Padhi, Y. Schiff, I. Melnyk, M. Rigotti, Y. Mroueh, P. Dognin, J. Ross, R. Nair, E. Altman, Tabular transformers for modeling multivariate time series, in: ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, IEEE, 2021, pp. 3565–3569. doi:10.1109/ICASSP39728.2021.9414142.

[59] X. Chen, N. Mishra, M. Rohaninejad, P. Abbeel, PixelSNAIL: An improved autoregressive generative model, in: International Conference on Machine Learning, PMLR, 2018, pp. 864–872. doi:10.48550/arXiv.1712.09763.

[60] P. Esser, R. Rombach, B. Ommer, Taming transformers for high-resolution image synthesis, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, IEEE, 2021, pp. 12873–12883. doi:10.1109/CVPR46437.2021.01268.

[61] A. Razavi, A. Van den Oord, O. Vinyals, Generating diverse high-fidelity images with VQ-VAE-2, Proceedings of the 33rd International Conference on Neural Information Processing Systems 32 (2019) 14866–14876. doi:10.48550/arXiv.1906.00446.

[62] A. Van Den Oord, N. Kalchbrenner, K. Kavukcuoglu, Pixel recurrent neural networks, in: Proceedings of the 33rd International conference on machine learning, Vol. 48, PMLR, 2016, pp. 1747–1756. doi:10.48550/arXiv.1601.06759.

[63] S. Black, S. Biderman, E. Hallahan, Q. Anthony, L. Gao, L. Golding, H. He, C. Leahy, K. McDonell, J. Phang, M. Pieler, U. S. Prashanth, S. Purohit, L. Reynolds, J. Tow, B. Wang, S. Weinbach, GPT-NeoX-20B: An Open-Source autoregressive language model, in: Proceedings of BigScience Episode – Workshop on Challenges Perspectives in Creating Large Language Models, Association for Computational Linguistics, Stroudsburg, PA, USA, 2022. doi:10.18653/v1/2022.bigscience-1.9.

[64] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901. doi:10.48550/arXiv.2005.14165.

[65] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al., Improving language understanding by generative pre-training (2018).

[66] Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, D. Roblek, O. Teboul, D. Grangier, M. Tagliasacchi, et al., AudioLM: a language modeling approach to audio generation, IEEE/ACM Transactions on Audio, Speech, and Language Processing 31 (2023) 2523–2533. doi:10.1109/TASLP.2023.3288409.

[67] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, Wavenet: A generative model for raw audio, arXiv preprint arXiv:1609.03499 (2016). doi:10.48550/arXiv.1609.03499.

[68] K. Ethayarajh, How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Association for Computational Linguistics, Stroudsburg, PA, USA, 2019. doi:10.18653/v1/D19-1006.

[69] Y.-L. Boureau, J. Ponce, Y. LeCun, A theoretical analysis of feature pooling in visual recognition, in: Proceedings of the 27th international conference on machine learning (ICML-10), 2010, pp. 111–118.

[70] A. Haviv, O. Ram, O. Press, P. Izsak, O. Levy, Transformer language models without positional encodings still learn positional information, arXiv preprint arXiv:2203.16634 (2022).

[71] Y. Takimoto, Y. Tanaka, T. Iwata, M. Okawa, H. Kim, H. Toda, T. Kurashima, Meta-learning for neural network-based temporal point processes, arXiv preprint arXiv:2401.15846 (2024).

[72] S. Xue, X. Shi, Z. Chu, Y. Wang, F. Zhou, H. Hao, C. Jiang, C. Pan, Y. Xu, J. Y. Zhang, Q. Wen, J. Zhou, H. Mei, EasyTPP: Towards open benchmarking the temporal point processes, in: International Conference on Learning Representations (ICLR), 2024. doi:10.48550/arXiv.2307.08097.