Efficient Transformers for Financial Data

Vladimir Baikalov [1] **Kovaleva Maria** [1] **Konstantin Shlychkov** [1] **Vo Ngoc Bich Uyen** [1]

Abstract

The attention-based methods and transformers made a significant breakthrough in the deep learning area and greatly impacted NLP task solutions. Although recent works show that they could potentially improve results in different task domains, the application of transformers for financial data in particular transaction data is unexplored. While applying attention mechanisms, one can face the apparent restriction on input sequence length due to the method's quadratic complexity. Recent papers proposed different ways to overcome this problem, but we want to concentrate on two promising approaches: Informer and Performer. The Informer is the most current and prospective approach. Its main assumption is that the model should have an "infinite memory" and fit a sequence with arbitrary length. The Performer model shows good results in the NLP task but is not well-explored for other datatypes. Its main idea is to use some trigonometric approximation of the attention matrix to decrease memory consumption. To sum up, the project aims to compare several recent methods proposed to decrease the evaluation complexity in particular tasks predicting the user's gender based on transactions.

1. Introduction

The most popular sequence transduction models are built on complicated recurrent or convolutional neural networks with an encoder and a decoder. Transformer is proposed as a new fundamental network design based solely on attention techniques for drawing global dependencies between input and output. Transformers are strong neural network designs that have emerged as the standard in many fields of machine learning. These models surpass others in terms of quality while being more parallelizable and taking substantially less time to train. Transformer-based language models have achieved impressive results by increasing the context size. However, whereas humans process information sequentially, continually updating their memories, and recurrent neural networks (RNNs) update a single memory vector across time, transformers do not - they comprehensively query every presentation connected with prior events. The regular Transformer scales quadratically with the number of tokens L in the input sequence, which makes huge datasets prohibitively expensive. Most approaches either limit the attention mechanism to local neighborhoods or add structural priors on attention. As a result, the amount of work they must perform rises in proportion to the length of the context. In this study, we present two possible solutions to this challenge: Informer and Performer.

First, we introduce the Informer or infinite-former - a transformer that extends the vanilla transformer with an unbounded long-term memory(LTM). By making use of a continuous-space attention framework that trades off the number of information units that fit into memory (basis functions) with the granularity of their representations. This representation has two significant advantages: (i) the context can be represented using a smaller number of basis functions N than the number of tokens, lowering the attention computational cost; and (ii) N can be fixed, allowing unbounded context to be represented in memory without increasing its attention complexity. When expressing the input sequence as a continuous signal, however, choosing a lesser number of basis functions results in reduced accuracy. To address the issue of resolution loss, we offer the notion of "sticky memories," which is a process that ensures the preservation of critical information in the LTM. In "sticky memories," we assign a bigger space in the LTM signal to sections of memory that are regularly examined, allowing the model to collect long concepts without losing significant information.

Second, we present the Performer, the first Transformer designs capable of provably accurate and practical estimation of regular (softmax) full-rank attention. Other strategies that try to reduce the space complexity of Transformers include reversible residual layers that allow for one-time activation storage in training and shared attention weights. Performer is the first linear structure that is fully compatible with conventional Transformers

[1]Skoltech University, Moscow, Russia. Correspondence to: Vladimir Baikalov <vladimir.baikalov@skoltech.ru>.

(with minor fine-tuning). Performer uses Fast Attention Via the positive Orthogonal Random features (FAVOR+) mechanism, leveraging new methods for approximating softmax and Gaussian kernels. FAVOR+ can also be used outside of the Transformer scope as a more scalable substitute for frequent attention.

Finally, in a specific task that predicts the gender of a customer using information about receipts and expenditures on a bank card, we execute three comparisons: baseline model, Performer model, and Informer model. The correctness of all models is then computed in terms of speed and memory usage.

2. Background

2.1. Vanilla Transformer

The model [1] follows the standard encoder-decoder architecture, but it omits the recurrent module and relies only on the attention mechanism to compute representations of its input and output. The encoder is composed of several layers of multi-head self-attention mechanism followed by a feed-forward layer, along with residual connections and layer normalization. The decoder has a similar structure but additionally uses the attention between the output from the previous decoder layer and the encoder output. To prevent the leftward information in decoder self-attention, the masking technique is used to avoid illegal connections.

The attention mechanism uses queries (Q), keys (K), and values (V) as its input and aims to estimate the contribution of each query for all values by measuring the similarity between keys and queries. The resemblance is measured with a standard scaled dot-product with further application of the softmax function to obtain the weights on the values. We can describe this operation as follows:

$$\text{Attention}\left(Q, K, V\right) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

where d is the dimension of hidden representations for queries and keys. Its usage may be illustrated with the following example: suppose ξ and η are independent random vectors, whose components have mean 0 and variance 1. Then their dot product (ξ, η) has mean 0 and variance d, which means the values may become large and hence result in computational instability. To avoid this we need to scale the dot product by $\sqrt{d}$.

To get more stable and generalized hidden representations the attention mechanism is separated into h groups. Queries, keys, and values are linearly projected into smaller subspaces with dimensions d_k, d_k, and d_v respectively, and then the attention mechanism is applied to each group. Obtained values are concatenated and projected again to result in the final result. This procedure may be represented as follows:

$$\text{MultiHead}\left(Q, K, V\right) = \text{Concat}\left(\text{h}_1, \text{h}_2, \ldots, \text{h}_z\right)W^l$$

where $\text{h}_i = \text{Attention}\left(QW_i^q, KW_i^k, VW_i^v\right)$ and $W_i^q, W_i^k \in \mathbb{R}^{d\times d_k}$, $W_i^v \in \mathbb{R}^{d\times d_v}$, $W^l \in \mathbb{R}^{d_v z\times d}$. This allows not only jointly attending to information from different representation subspaces but also speeding up the computation with parallelization.

However, there is an enormous limitation in computational complexity as the attention mechanism requires all pairwise dot products between two inputs and the model works extremely slowly when handled with long sequences. Many approaches were proposed to overcome this issue, but we will investigate only two of them — performer and ∞-former.

2.2. ∞-former

Transformers are unable to model long-term memories effectively since the amount of computation they need to perform grows with the context length. All efficient transformers have a finite memory capacity and are forced to drop old information. The ∞-former [2] is a transformer, which extends the vanilla transformer with unbounded long-term memory. By making use of a continuous space attention mechanism to attend over the long-term memory, the ∞-former's attention complexity becomes independent of the context length, trading of memory length with precision. In order to control where precision is more important, ∞-former maintains "sticky memories," being able to model arbitrarily long contexts while keeping the computation budget fixed.

The main idea of this model is the usage of continuous attention using N radial basis functions (RBF).

Let's consider the details of using continuous attention. $X = [x_1, ..., x_L]$ - input vector. Every x_i is associated with $t_i \in [0, 1]$, so the input vector can be represented as $\bar{X}(t) = B^T\psi(t)$, where $\psi(t)$ is a vector of N RBFs: $\psi_j(t) = N(t; \mu_j, \sigma_j)$ and B is a coefficient matrix, which can be found by ridge regression: $B^T = X^T F^T \left(FF^T + \lambda I\right) = X^T G$, where $F = [\psi(t_1), ...\psi(t_L)]$. As out input vector X is replaced by $\bar{X}(t) = B^T\psi(t)$ so the keys and values will be computed as $K_h = B_h W^{K_h}$ and $V_h = B_h W^{V_h}$ (where different h associated with different heads). And for every query $q_{h,i}$ we need to compute the parameters (which are given by neural network) of the normal distribution from which we will sample our long-term memory (LTM) context vectors: $\mu_{h,i} = \text{sigmoid}\left(\text{affine}\left(\frac{K_h q_{h,i}}{\sqrt{d}}\right)\right)$, $\sigma_{h,i} = \text{softplus}\left(\text{affine}\left(\frac{K_h q_{h,i}}{\sqrt{d}}\right)\right)$, $p_{h,i} \sim N\left(t; \mu_{h,i}, \sigma^2_{h,i}\right)$. And having value function $\bar{V}_h(t) = V_h^T \psi(t)$ we will have $Z_{h,i} = \mathbb{E}_{p_{h,i}}\left[\bar{V}_h\right] = V_h \mathbb{E}_{p_{h,i}}\left[\bar{\psi}(t)\right]$ which forms the rows of matrix $Z_{\text{LTM},h}$. $Z_{\text{LTM}} = \left[Z_{\text{LTM},1}, ..., Z_{\text{LTM},H}\right] W^o$ is LTM context representation. So the final context representations in this framework will be $Z = Z_T + Z_{\text{LTM}}$, where Z_T is e transformer context vector.

The key matrix K_h size depends only on the number of basis functions N, but not on the length of the context being attended to. Thus, the ∞-former's attention complexity is also independent of the context's length. It corresponds to $O(L^2 + LN)$, while the complexity of a vanilla transformer attending to the same context is $O(L(L + L_{\text{LTM}}))$

When representing the memory as a discrete sequence, to extend it, we need to store the new hidden states in memory. In a vanilla transformer, this is not feasible for long contests due to the high memory requirements. However, the ∞-former can attend to unbounded context without increasing memory requirements by using continuous attention. To be able to build an unbounded representation, we first sample M locations in $[0, \tau]$, where $\tau < 1$, and evaluate $\bar{X}(t)$ at those locations. Then, we concatenate the corresponding vectors with the new vectors coming from the short-term memory $X = \left[X_{\text{past}}^T, X_{\text{new}}^T\right]^T$. Finally, we simply need to perform multivariate ridge regression to compute the new coefficient matrix B, via $B^T = X^T G$, where $G \in \mathbb{R}^{(M+L)N}$.

Also, the sticky memories concept was proposed by authors of ∞-former. The main idea of this concept is not to use linearly spaced M locations, but to sample them according to histogram based on the attention given to each interval of the signal on the previous step.

2.3. Performer

Computation of attention matrix in vanilla transformer architecture requires quadratic space and time complexity with number or tokens/sequence length and it does not rely on any priors such as data sparsity or specific data structure. Paper *Rethinking attention with performers* [3] proposes a proven way for accurate estimation of regular (softmax) full-rank attention, but with linear space and time complexity. Performers use the above-mentioned mechanism called FAVOR+. More precisely, the authors of this paper show that it is possible to approximate attention matrix A having $O(Ld^2 \log(d))$ time complexity.

The main idea with is the basis of this paper states that it is possible to reconstruct in practice most kernels ϕ in the following way:

$$\phi(x) = \frac{h(x)}{\sqrt{m}} \left(f_1(\omega_1^\top x), \ldots, f_1(\omega_m^\top x), \ldots, f_l(\omega_m^\top x)\right)$$

where functions $f_1, \ldots, f_l : \mathbb{R} \to \mathbb{R}$, function $g : \mathbb{R}^d \to \mathbb{R}$ and every $\omega_i \overset{\text{iid}}{\sim} \mathcal{P}(\mathbb{R}^d)$ is a deterministic vector. For example with $h(x) = 1$, $l = 2$, $f_1 = \sin$, $f_2 = \cos$ corresponds to shift-invariant kernels, and with $\mathcal{P}(\mathbb{R}^d) = \mathcal{N}(0, \mathbf{I}_d)$ we can get Gaussian kernel.

The key *softmax-kernel* itself may be written in the following form:

$$SM(x, y) \overset{\text{def}}{=} \exp\left(x^\top y\right) = \exp\frac{\|x\|^2}{2} K_{gauss}(x, y) \exp\frac{\|y\|^2}{2}$$

This equation can be estimated by using $h(x) = \exp\frac{\|x\|^2}{2}$, $l = 2$, $f_1 = \sin$, $f_2 = \cos$ and it can be called $\hat{SM}_m^{trig}(x, y)$. But since $\sin$ and $\cos$ may get negative values it may lead to high variance and abnormal behaviors.

To tackle this issue author proposes two new estimators:

$$h(x) = \exp\left(-\frac{\|x\|^2}{2}\right)$$

with

$$l = 1, f_1(u) = \exp(u)$$

$$h(x) = \frac{1}{\sqrt{2}} \exp\left(-\frac{\|x\|^2}{2}\right)$$

with

$$l = 2, f_1(u) = \exp(u), f_2(u) = \exp(-u)$$

which is called $\hat{SM}_m^+$ and $\hat{SM}_m^{hyp+}$ respectively. For both estimators $\mathcal{D} = \mathcal{N}(0, \mathbf{I}_d)$.

Also, the authors show that orthogonal choice of all random features ω will lead to better results.

To sum up, authors of *Rethinking attention with performers* [3] presented models called *Performer*, a new type of Transformers, relying on our Fast Attention Via positive Orthogonal Random features (FAVOR+) mechanism to significantly improve space and time complexity of regular Transformers. Their mechanism provides the first effective unbiased estimation of the original softmax-based Transformer with linear space and time complexity.

3. Conducted experiments

3.1. Experiments setup

For metrics evaluation, we used a publicly available dataset with sequences of users' transactions. It includes more than 3.5 million transactions for about 8.5 thousand users. The transaction includes its time and amount with the corresponding mcc code and their types. Additionally, there are descriptions for each mcc code and transaction type. Based on this information we need to predict the gender of user.

For data prepossessing, we used logarithmic scaling for the amount to prevent gradients explosion and large variance in data as initial values are large in modulus. Additionally, we used feature engineering with extra statistics (min, max, mean, etc) for amount sequences to provide more information for models.

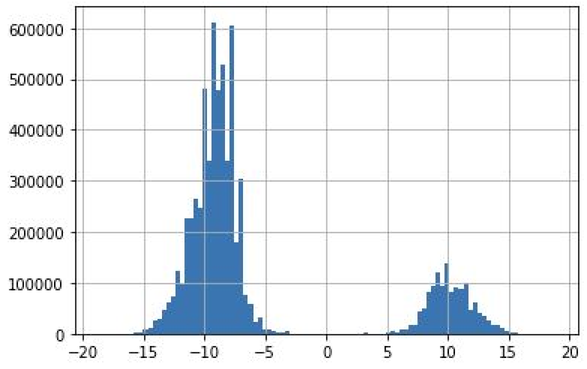

Figure 1. Amount Distribution After Scaling

We limited all sequences larger than 1024 to avoid extremely long ones as more than 90% users have transaction history no bigger than this threshold. After all, we padded sequences until the proposed threshold to align them. This prepossessing step was done for both mcc codes and corresponding types of operations.

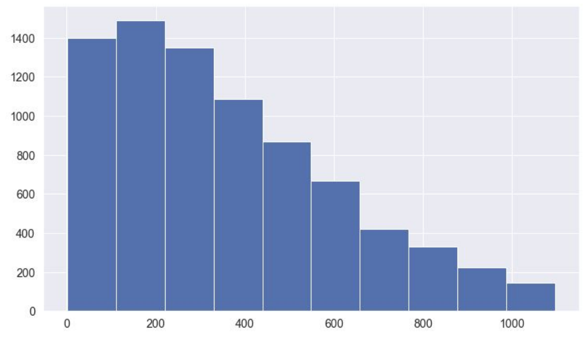

Figure 2. Length Distribution of Transaction Sequences

Finally, all sequences were sorted by their corresponding times. So, for each user, we got a 1024-long sequence for both mcc codes and types of operation and a vector with 8 statistics for the amount.

3.2. Models setup

For fair comparison we used a standard transformer with different variations of attention layer:

- softmax-attention
- informer-attention
- performer-attention

We extended the transformer with an additional projection block, that proceeds input (three tokens sequences — mcc codes, their types, positions and one vector with statistics for an amount field) into one vector. It maps each token into a vector with an embedding layer to obtain embeddings for each of the three sequences and additionally projects the vector with statistics into the necessary dimension. Then all four embeddings are summed into one to get input for the transformer layer. The outputs from the transformer are further projected into the desired dimension (in our case — 1). So, the overall architecture of our models may be described as follows:

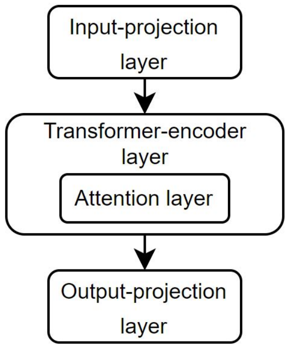

Figure 3. Scheme of Proposed Model

For all models, we used the same hyperparameters setup: the number of layers for the encoder was set to 2, the number of heads for attention — 4, and the hidden dimension for embeddings — 128.

3.3. Training setup

All experiments were handled on a single GPU with 12GB memory. All models were trained for 30 epochs and for optimization we used Adam with initial learning rate $3 \cdot 10^{-4}$ and StepLR scheduler with $\gamma = 0.8$ and step = 5. A cross-entropy was taken as a loss function. For quality evaluation, we used standard classification metrics —- accuracy, recall, precision, F1 score, and roc-AUC score. The data was split in a ratio of $4 : 1$ for train and test respectively.

4. Results of experiments

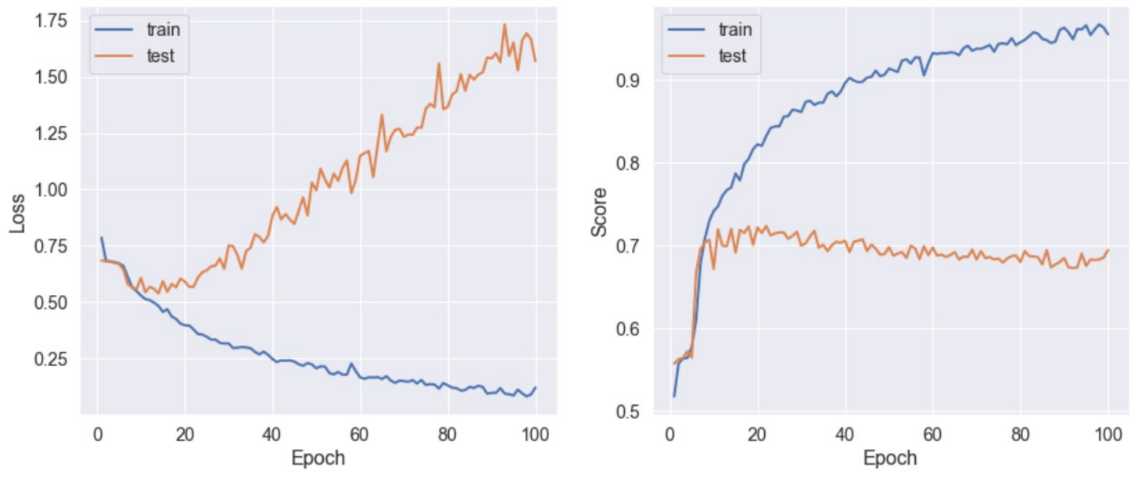

Figure 4. Loss and accuracy scores during training got baseline model

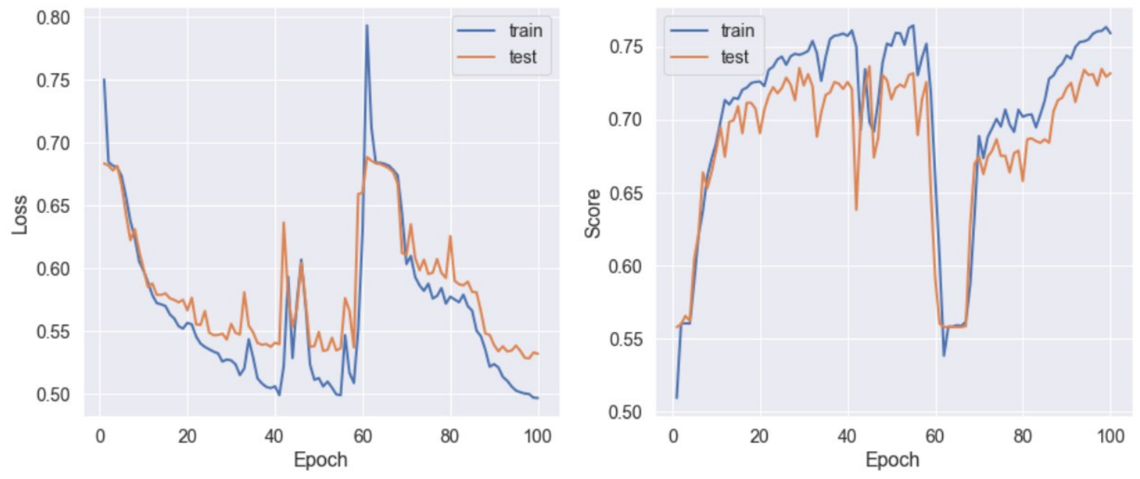

Figure 5. Loss and accuracy scores during training got performer model

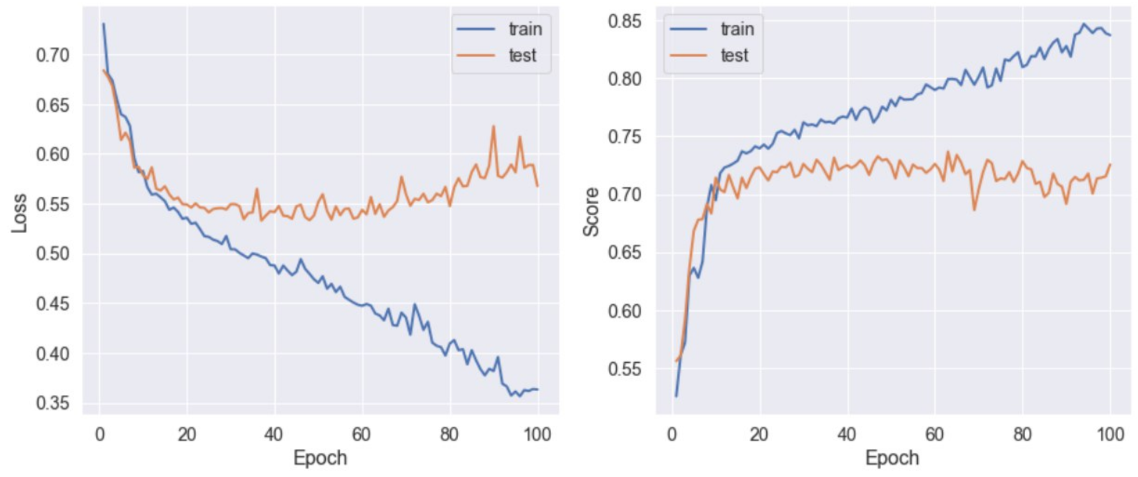

Figure 6. Loss and accuracy scores during training got informer model

Model	Time per 1 epoch (100 epochs)	Train score	Validation score
Baseline	1.68s	95.51	69.46
Performer	1.91s	75.89	73.15
Informer	2.73s	83.74	72.56

All the models achieve at least 70% accuracy score on the test data. Figure 5 depicts the Performer model at its most accurate. Figures 4 and 6 depict typical overfitting models, although the baseline and Informer models can exhibit better outcomes with additional data or with minor modifications.

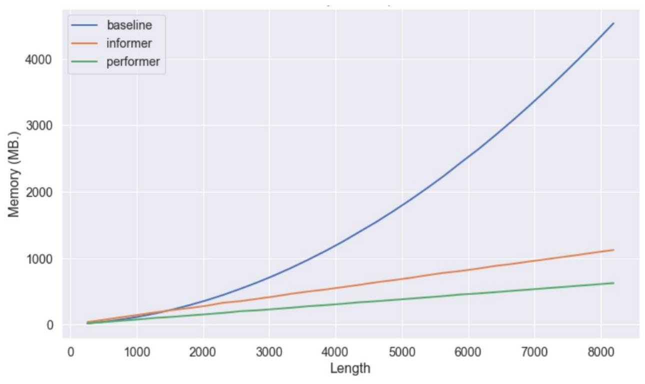

Figure 7. Memory consumption

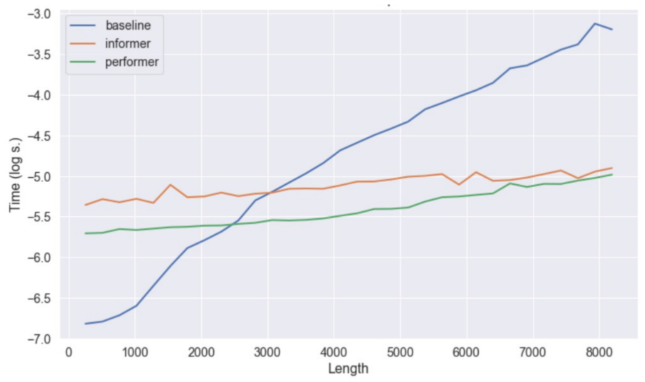

Figure 8. Inference time comparison

Figures 7 and 8 show that models with Performer and Informer attention layers work faster and require less memory than model with regular multihead attention. While memory and time consumption grows quadratically from the sequence length for the baseline model, models with Informer and Performer layers consume memory and time linearly.

5. Conclusion

In this work, we presented three types of transformers:

- a sequence transduction model used in encoder-decoder architectures with multi-headed self-attention - Vanilla transformers;
- a transformer extended with an unbounded long-term memory - Informer;
- a transformer relied on Fast Attention Via positive Orthogonal Random features mechanism - Performer.

Experiments on transaction data to determine a customer's gender, show the comparisons of the memory and compute efficiency of these models. We can conclude that the Performer and Informer model overcome the memory and temporal complexity significantly, which rise quadratically from the baseline model's sequence length.
The code we used to train and evaluate our models is available at https://github.com/NonameUntitled/MSDProject

References

1. Attention is all you need / A. Vaswani [et al.] // Advances in neural information processing systems. — 2017. — Vol. 30.
2. *Martins P. H.*, *Marinho Z.*, *Martins A. F.* ∞-former: Infinite Memory Transformer // arXiv preprint arXiv:2109.00301. — 2021.
3. Rethinking attention with performers / K. Choromanski [et al.] // arXiv preprint arXiv:2009.14794. — 2020.

Project plan

Milestones	Doer	Deadline
Study materials and write first report		03/12/2022
1.1. Full attention model + Abstract	Konstantin Shlychkov	
1.2. Performer model	Vladimir Baikalov	
1.3. Informer model	Kovaleva Maria	
1.4. Introduction + Project plan	Vo Ngoc Bich Uyen	
Implementation for:		16/12/2022
2.1. Baseline model	Vladimir Baikalov	
2.2. Performer model	Konstantin Shlychkov	
2.3. Informer model	Kovaleva Maria	
Benchmarking all models (speed and memory consumption) Comparing three models and concluding experiments	Konstantin Shlychkov	17/12/2022
Write final project report	All team members	17/12/2022
Make a presentation	All team members	17/12/2022