Anomaly Detection in Time Series Data: Gambling prevention using Deep Learning

Bibiána Lajčinová [1], Marián Gall [1,2] and Michal Pitoňák [1,3]

[1]Slovak National Supercomputing Centre, Dúbravská cesta 3484/9, 84104 Bratislava, Slovak Republic
[2]Institute of Information Engineering, Automation and Mathematics, Faculty of Chemical and Food Technology, Slovak University of Technology in Bratislava, Radlinského 9, 812 37 Bratislava, Slovak Republic
[3]Department of Physical and Theoretical Chemistry, Faculty of Natural Sciences, Comenius University in Bratislava, Ilkovičova 6, 842 15 Bratislava, Slovak Republic

Abstract

Gambling prevention of online casino players is a challenging ambition with positive impacts both on player's well-being, and for casino providers aiming for responsible gambling. To facilitate this, we propose an unsupervised deep learning method with an objective to identify players showing signs of problem gambling based on available data in a form of time series. We compare the transformer-based autoencoder architecture for anomaly detection proposed by us with recurrent neural network and convolutional neural network autoencoder architectures and highlight its advantages. Due to the fact that the players' clinical diagnosis was not part of the data at hand, we evaluated the outcome of our study by analyzing correlation of anomaly scores obtained from the autoencoder and several proxy indicators associated with the problem gambling reported in the literature.

Introduction

Gambling prevention of players with problem or pathological gambling, currently conceptualized as a behavioural pattern where individuals stake an object of value (typically money) on the uncertain prospect of a larger reward [1], [2], is of high societal importance. Research over the past decade has revealed multiple similarities between pathological gambling and the substance use disorders [3]. With the high accessibility of the Internet, the incidence of pathological gambling has increased. This disorder can result in significant negative consequences for the affected individual and his/her family too. Therefore detecting early warning signs of problem gambling is crucial for maintaining player's well-being.

This work is a joint effort of Slovak National Competence Center for High-performance Computing, DOXXbet, ltd. – sports betting and online casino, and Codium, ltd. – software developer of the DOXXbet sports betting and iGaming platform, with the goal to enhance customer service and players' engagement via identification and prevention of gambling behaviour. This proof of concept is a foundation for future tools, which will help casino mitigate negative consequences for players, even for a price of less provision for the provider, as in line with European trends in risk management related to problem gambling.

There has been a lot of research focusing on real money gambling with the use of machine learning methods. Most of the studies work with players' diagnostic assessment which makes use of supervised learning possible. In the absence of clinical diagnoses and indicators of players' financial problems within the dataset, researchers must rely only on proxy indicators amongst which voluntary self-exclusion is the most common. Other symptoms mentioned in the literature include account closure, complaints to the online casino provider, requests to increase spending limits [4], chasing losses by gambling more, instant gratification (a behaviour when a player prefers to get immediate but smaller reward than delayer bigger reward) and many others [5], [6]. From behavioural data only, it is not possible with absolute certainty to tell if the player has a gambling problem. Despite this, our approach attempts to detect early warning signs of gambling disorder in an unsupervised manner and then analyze these symptoms and other anomalous actions.

One of the recent studies focused on identifying potential gamblers in skill-based cash games [7] used a combination of long short-term memory (LSTM) neural network layers and adversarial autoencoder (AE). Authors of this paper utilize information related to players' clinical diagnosis, therefore do not have to rely on side symptoms / proxy indicators. Another study [8] uses objective of segmenting users of online gambling platforms according to the tendency of these users to have compulsive gambling behaviour. This paper works with unlabelled data.

In our study we propose a completely unsupervised deep learning approach using transformer-based AE architecture to detect anomalies in the dataset - players with anomalous behaviour. The dataset at hand does not comprehend the clinical diagnosis, and amongst other proxy indicators mentioned before only few are available - requests to increase spending limits, chasing losses by gambling more (referred to as chasing episodes later in this article), usage of multiple payment methods, frequent withdrawals of small amount of money and other mentioned later in the text. Clearly, not all the anomalous users must necessarily have problem gambling, hence the proxy indicators are used in combination with AE results, namely the anomaly score. The foundation of our approach rests on the idea that a compulsive gambler is an anomaly within the active casino players, with the literature mentioning their fraction amongst all players being between 0.5% to 5% for chance-based games [7].

Data

The data acquired for this research consist of sequences of data points collected over time, tracking multiple aspects of player's behaviour such as frequency and timing of their gaming activities, frequency and amount of cash deposits, payment methods used when depositing cash, information about the bets, wins, losses, withdrawals and requests for change of deposit limit.

Feature engineering resulted in 19 features in a form of time series (TS), so that each feature consists of multiple time stamps. These features can be classified into three categories - "time", "money" and "despair", as inspired by Seth *et al.* [7]. Table 1 summarizes the full set of TS features with a short explanation.

Each feature is a sequence of N values, where each value stands for one out of N consecutive time windows. This value was produced by aggregating daily data in the respective time window, with the time window length being specified in the Table 1 together with the information about the time window being sliding or not. Hence, for each sample we needed a history of N time windows. Feature engineering procedure is displayed in Figure 1 and the final data shape is depicted in Figure 2.

The length of the sequence N is an important factor in prediction quality. While a larger N generally yields better results, increasing N is in contradiction with our goal to detect anomalies at the earliest possible time. Thus, smaller value of N is preferred. After a set of experiments a series length of 8 was chosen. Three types of aggregating function were used to create the features: maximum, minimum or sum.

Let us provide more detailed explanation for selected features from Table 1:

- *Money 10* - Total number of days when a player did at least two small withdrawals. This refers to number of days, when a player withdrew money at least twice a day and the sum of the withdrawal amounts was <= 30th percentile of the withdrawal amount sums for players doing frequent withdrawals (at least two withdrawals per day).
- *Chasing ep. 1 & 2* - Number of days in a time window when player chased the loss. Chasing episode event happens when a player has a peak of losses (local maximum within a time window) accompanied by a peak of at least three cash deposits.

See a randomly selected sample (set of 19 TS for one player, each series representing one feature) in Figure 3.

Data cleaning

Identifying and removing outliers/abnormal data points from the training dataset when work-

Context	Feature	Time frame	Description
Time	Time 1	15 day SW	The max. number of games played on a day during a time window.
	Time 2	15 day SW	Total number of weekend games during a time window.
	Time 3	15 day SW	Total number of weekday games during a time window.
	Time 4	15 day SW	Total number of games between 12AM - 6AM during a time window.
	Time 5	15 day SW	Total number of logins during a time window.
Money	Money 1	1 month SW	Net loss incurred in last one month.
	Money 2	3 months SW	Net loss incurred in last three months.
	Money 3	15 day SW	Net loss incurred during a time window.
	Money 4	15 day SW	The max. number of times cash was added in a single day during a time window.
	Money 5	15 day SW	Number of multiple (> 3) single-day add cash transactions during a time window.
	Money 6	15 day SW	The max. amount of cash that was added in a single day during a time window.
	Money 7	15 day SW	Maximum number of different payment modes used in a day over a time window.
	Money 8	15 day SW	Number of days in a time window when more than two payment modes were used.
	Money 9	15 day SW	Total number of money withdrawals during a time window.
	Money 10	15 day SW	Total number of days when a player did at least two small withdrawals.
Despair	Despair 1	15 day SW	Min. win ratio in a day over a time window, to detect a bad streak of games.
	Despair 2	15 day SW	Num. of times in a time window the player tried to increase his daily deposit limit.
	Chasing ep. 1	15 day SW	Number of days in a time window when player chased the loss.
	Chasing ep. 2	5 day TW	Number of days in a time window when player chased the loss.

Table 1: *List of time-series features used in all of the models. Abbreviations in Time frame columns mean: SW stands for sliding window and TW stands for time window - in this case the time windows are disjunctive.*

ing with AE models for anomaly detection is an important step to ensure the model learns dominantly on non-anomalous data points [9]. The dataset at hand presumably does contain anomalous samples, since it is a collection of TS data for all casino players, including suspect gamblers.

Our approach for dataset cleaning is based on the idea of using a *rule engine*. A rule engine is a set of rules that can be utilized as a simple method aimed at detecting anomalous players. These rules are based on proxy indicators, namely the number of requests for limit changes, mean number of players' logins and withdrawals and mean number of players' chasing episodes. In order to apply these rules, we set the thresholds at the 95th percentile of the values for each respective proxy indicator. If a data point satisfies at least three out of these four rules, it is classified as an outlier and subsequently removed from the training set. Altogether 211 samples were removed, representing slightly less than 1% of the overall dataset size.

Data normalization

To improve the predictive quality of the model even further the data was normalized after the the feature engineering and cleaning steps. Normalization facilitates more balanced contribution of features within the model. *Z-Score normalization* was chosen after several experiments with other scaling techniques such as *log* and *minmax*.

Z-Score normalization was carried out as follows: Let $\mathbf{x} \in \mathbb{R}^D$ be a vector $\mathbf{x} = (x_i, \ldots, x_D)$. We first compute the mean and the standard deviation of $\mathbf{x}$:

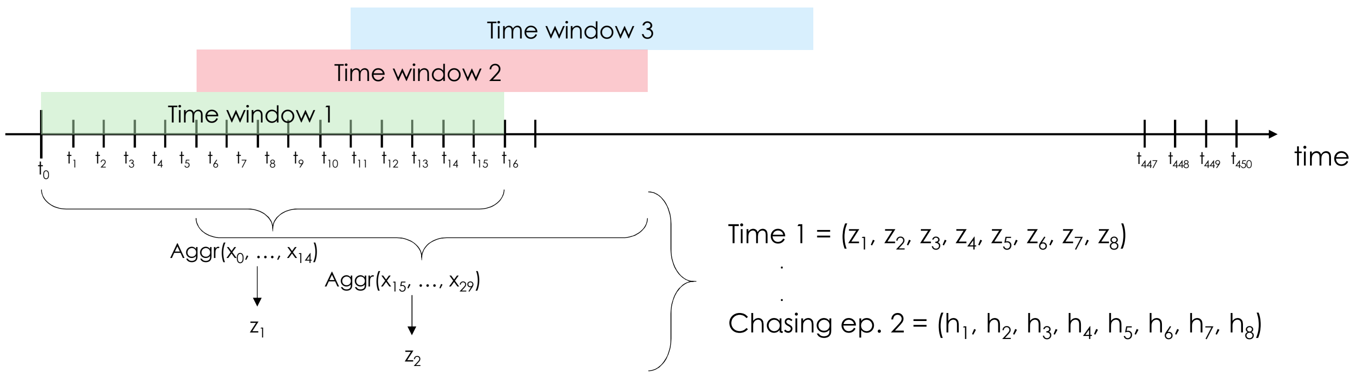
$$\mu_x = \frac{1}{D}\sum_{i=1}^{D} x_i \tag{1}$$

$$\sigma_x = \sqrt{\frac{1}{D}\sum_{i=1}^{D}(x_i - \mu_x)^2} \tag{2}$$

The z-score normalization of $\mathbf{x}$ is then calculated as:

$$ZN(x) = \frac{\mathbf{x} - \mu_x \mathbf{1}}{\sigma_x} \in \mathbb{R}^D \tag{3}$$

where $\mathbf{1} = [1, \ldots, 1]^T$ is a D-dimensional vector of ones. This normalization has a property of



Time window 3
Time window 2
Time window 1
time
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16}
t_{447} t_{448} t_{449} t_{450}
Aggr(x_0, ..., x_{14})
z_1
Aggr(x_{15}, ..., x_{29})
z_2
Time 1 = (z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8)
Chasing ep. 2 = (h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8)

Figure 1: *Visualization of the data aggregation from daily basis into time windows, and eventually to TS features. $t_1, ..., t_{450}$ represent time stamps for daily data $x_1, ..., x_{450}$. Daily data points from a time window are aggregated into a single value z_i for all $i \in (1, \dots, 8)$.*

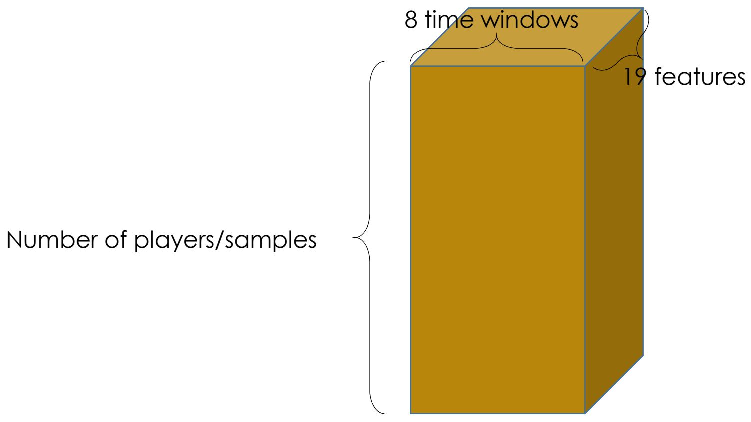


Figure 2: *Final data shape obtained after feature engineering. Each sample is represented by 19 features consisting of 8 time windows.*

transforming outliers in the dataset into values that are more standardized and comparable to the rest of the data, making it easier to analyse and interpret the distribution of the data.

For certain features that exhibited high variance among players, scaling was done on a per-player basis, while the remaining features, which showed low variance, were scaled on a per-dataset basis. This hybrid normalization proved to perform better than normalizing all samples per player or per whole dataset. Features scaled per-dataset are *despair 1* and *despair 2*.

Data clipping

Data clipping is another technique used in the study in addition to Z-Score normalization. Since the data provided for the study is completely unlabelled and only 211 samples were removed from the dataset by data cleaning, it is necessary to eliminate the impact of the remaining outliers by clipping. The clipping was done using a metric *Median Absolute Deviation* (MAD) defined in equation 4.

$$MAD = median(|x_i - \tilde{x}|) \tag{4}$$

where $\tilde{x}$ represents the median of the set of values $x_1, x_2, \dots, x_d$. MAD measures the average distance between each data point and the median of the dataset. If for a certain value applies

$$|x_i| > K * MAD \tag{5}$$

where K is a constant, than x_i is considered to be an outlier and therefore is clipped like:

$$x_i = x_i - MAD \tag{6}$$

Chosen value for K was 4.

Model architecture

TS is a set of observations that have been recorded in an orderly fashion and which are correlated in time. TS data can be used for forecasting of the future values, but also for anomaly detection, which is an active area of research. Deep learning is certainly not the only approach applicable on TS data, methods such as STL (seasonal-trend) decomposition [10] or regression trees can be applied as well.
The so-called autoencoder is an unsupervised Deep learning technique suitable for anomaly detection for TS data. The idea behind using

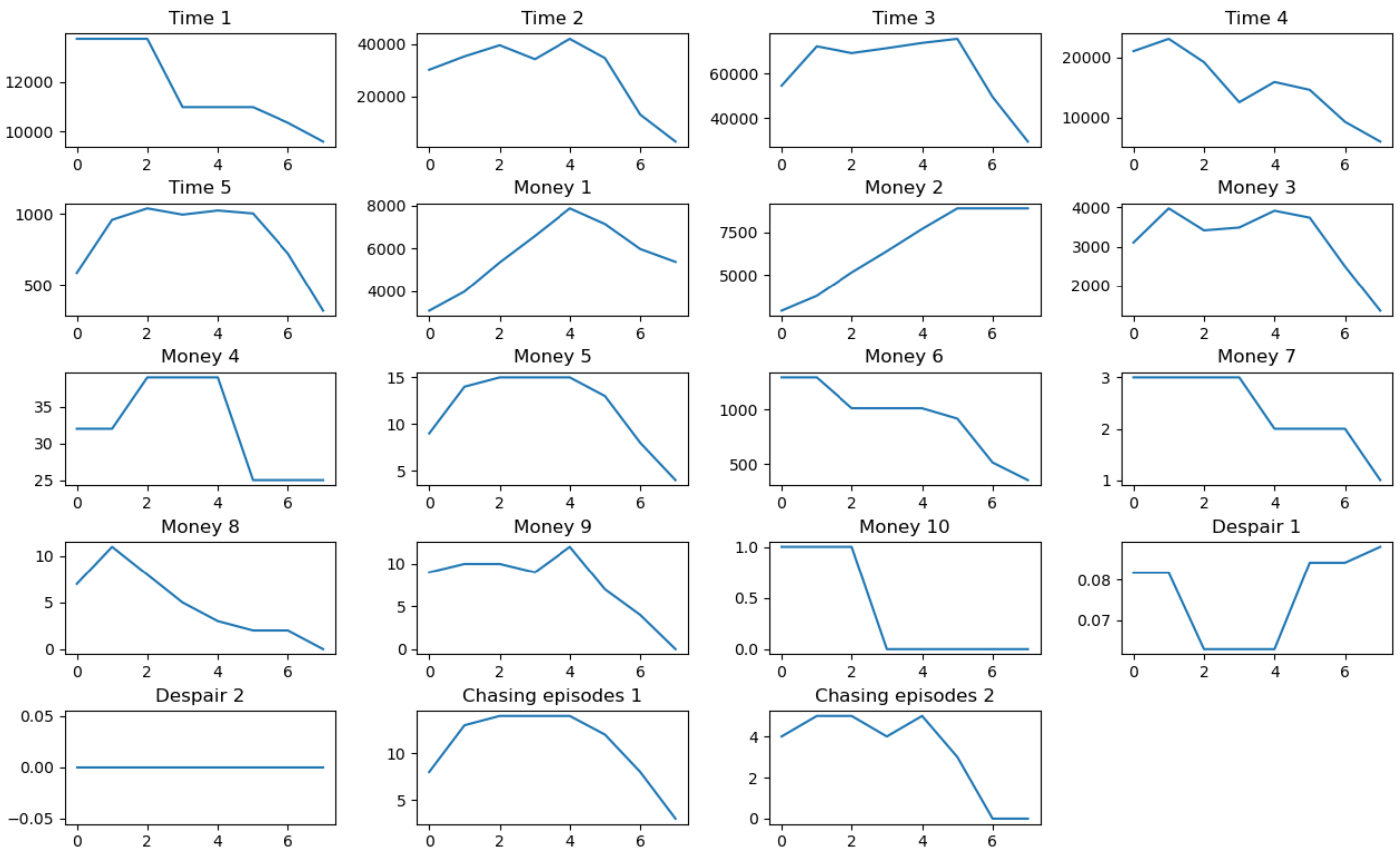


Figure 3: *Example of a data point. Each data point contains TS for 19 features. Each feature consists of 8 time stamps, each representing one 15-day long time window. Values shown are not scaled, neither clipped, but undergo these transformations prior to entering the model.*

this type of neural network to detect anomalies in data is based on reconstruction ability of the model. The AE model learns to reconstruct the data in the training set and since the training set is expected to contain mostly non anomalous data points the model learns to reconstruct well only samples that are *normal*. Therefore, when the input data sample is anomalous, trained AE model cannot reconstruct this data point *well enough* which results in high reconstruction error. Reconstruction error can be used to as anomaly score of a data sample, where a higher score indicates a greater likelihood of the sample being an anomaly.

AEs can further be split into three categories according to what type of layers/architecture they utilize [11].
First category is the LSTM-based neural network. LSTM stand for long-short term memory and it is a type of recurrent neural network which is well suited for sequential data. Another commonly used neural network architecture in CNN-based. CNN stands for convolutional neural network. In this approach a 1D convolutional layers is applied on TS sequence or the TS is transformed into 2D or 3D array and then use multidimensional convolutional layers.
Lastly, the transformer-based architecture is a novel class of deep learning models that were originally developed for natural language processing (NLP), but have successfully been applied to sequential data / TS analysis too [12].
LSTM and CNN network architectures are both widely used when working with TS data, but both of them have some drawbacks when compared to transformer-based architectures. LSTM and CNN architectures are not designed to capture long-range time dependencies in the data, which can limit their ability to make accurate predictions. Also these two architectures are primarily designed to model sequential relationships, therefore they do not capture non-sequential relationships (*i.e.* dependencies between two data points, that are no adjacent to each other) in the

data well, unlike transformers [13].
For every type of AE architecture there exists an adversarial version. Adversarial autoencoder (AAE) can turn an AE into a generative model. In this approach, AE is trained with two objectives, first one being a traditional reconstruction error criterion, and second an adversarial training criterion [14]. AAEs are also used in anomaly detection tasks.

Our approach

Usage of AAE architecture in analogous way as in [7] has proven not to be suitable for the data at our disposal.The reason is that the AAEs (as well as generative adversarial networks) in order to learn to detect anomalies using the generative AE and discriminative discriminator needs to be trained strictly only on normal data samples [15], [16]. Our dataset, as already mentioned, might (and almost certainly) does contain anomalous samples.

However, in order to compare results from our study with the findings in [7], we used the AE architecture proposed in the article, but without the discriminator component. Let's call this model *LSTM A*. Another two AE architectures we applied is an architecture leveraging LSTM layers as described in the article [17] (*LSTM B* model) and an architecture using 1-D convolutional layers (*CNN* model). Lastly, we trained AE with architecture based on transformers denoted as *Transformer*.

In the transformer-based architecture, both encoder and decoder contain *Multi Head Attention* layer with four heads and 32 dimensional keys and values vectors. This self-attention layer is followed by Feed Forward Network with dropouts and residual connections between them. The whole AE model has slightly more than 100000 trainable parameters. We have done some experiments with increasing the number of trainable parameters, but this number has proven to be sufficient, so the results of other experiments are not be presented here. Transformer-based AE architecture is visualised on Figure 4. One of the primary sources of inspiration for our model architecture is the code example from Keras blog [18]. This blog post presents an architecture for a transformer model for TS classification that caught our attention due to its effective utilization of self-attention mechanisms.

Model name	Number of parameters
LSTM A	2 435
LSTM B	691 163
CNN	291 299
Transformer	100 951

Table 2: *Number of trainable parameters of four evaluated AE models.*

Architectures for *LSTM A* and *LSTM B* models are described in the respective references. *CNN model* architecture was produced by us. Both encoder and decoder contain 5 1-dimensional convolutional layers followed by *batch normalization*, *dropout* and *maxpooling* layers, with 256, 128, 64, 32 and 8 filters in the encoder convolutional layers (and reversed number of filters in decoder layers). The total number of parameters for each of four AE models considered in this work is summarized in Table 2.

AE models comparison

All of the models were trained on the same dataset under the same conditions (Adam optimized, Mean Square Error (MSE) loss function). The dataset contains slightly more than 22,000 samples (with each sample consisting of 19 TS features). Some players are in the dataset multiple times, because their history of online activity is long enough to create several TS of length 8. When this happens, those multiple samples of one player are treated as completely independent.

About 10% of randomly selected data was used for validation and 10% for testing. The data was preprocessed identically for all four models (with cleaning, clipping and z-score normalization).

Training curves The history of training for all four our models is presented in Figure 5. Blue curves represent models' performance on training set and orange curves performance on validation set. On the x-axis is the number of epochs used for training and on the y-axis is the value of MSE loss.

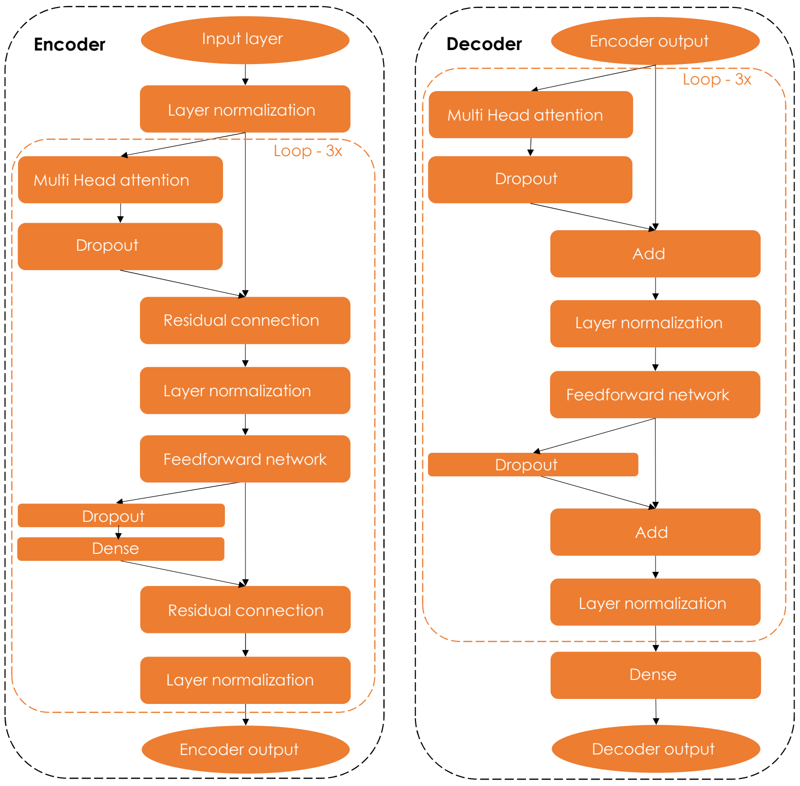


Figure 4: *Visualization of the transformer-based AE architecture.*

For all models the MSE loss drops rapidly at the beginning of training. The *transformer-based* model converged the most quickly amongst of all the models, approaching values very close to zero without being overfitted. *LSTM A* and *CNN* models do not show signs of overfitting, only *LSTM B* model begun to overfit at around 10th epoch. As for the MSE loss values reached at the end of training, they are the lowest for *Transformer*, followed by *LSTM B* model.

Reconstruction loss We performed a 3-fold cross-validation by splitting the data into training, validation, and test sets, and trained the models for each split to assess their stability. Resulting average loss values and their variances are displayed in the Table 3. The average reconstruction error of *Transformer* model is significantly lower than all the other models. *LSTM B* model comes second in the reconstruction performance and *CNN* model seems to have the worst prediction performance. Generally, the test loss is observed to be always higher than train and validation losses. The reason for this is that those 211 data points that were removed from the training set in the data cleaning process, were moved to the test set. Without moving these samples, the test loss for transformer-based model would be as low as 0.012, for CNN model 0.33, for LSTM A model 0.27, and for LSTM B model 0.13.

Table 3: *Performance of four AE models on training, validation and test dataset evaluated by cross-validation expressed in average MSE loss in the first row and variance of MSE in the second row.*

Model name	**Train**	**Valid.**	**Test**
LSTM A AE	0.258	0.269	0.279
Variance	4.2E-05	1.1E-05	1.4E-05
LSTM B AE	0.104	0.126	0.137
Variance	1.4E-06	3.6E-06	1.2E-05
CNN AE	0.316	0.325	0.343
Variance	9.8E-05	1.1E-05	1.1E-04
Transformer-based AE	0.010	0.012	0.015
Variance	3.5E-06	1.4E-06	5.0E-06

More detailed overview of the models' performance is displayed on the Figure 6 as histograms of loss values of the test set. All histograms have heavy right tail, which is expected for datasets containing anomalies. Similarly to the Figure 5, the transformer-based model exhibits significantly lower reconstruction loss values (x-axis).

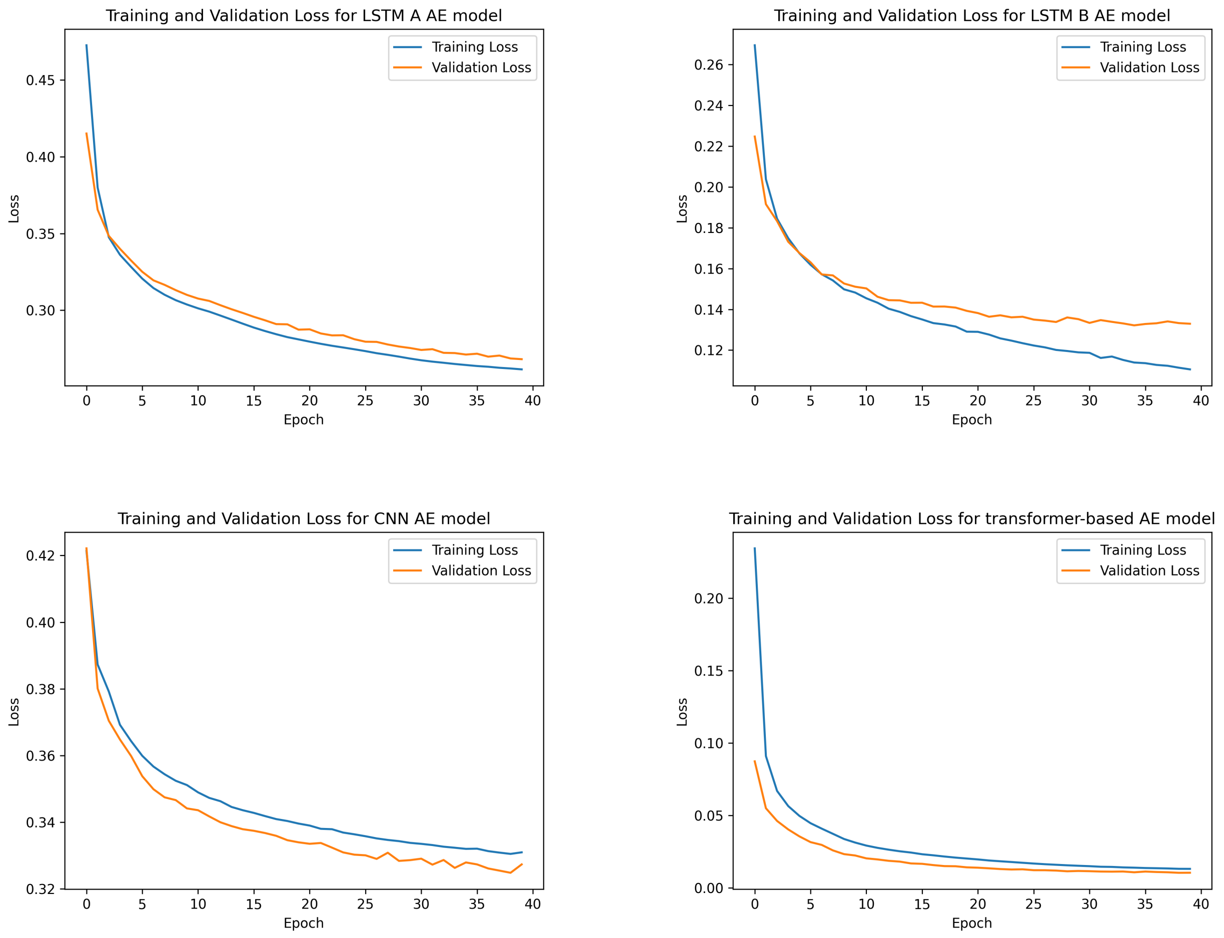


Figure 5: *Training curves for four AE architectures. The x-axis represents the number of epochs and y-axis the value of MSE (Mean square error) loss.*

Prediction ability To get a visual impression of how well is a TS reconstructed by all four evaluated models, Figure 8 depicts original (blue line) and predicted data (red line) for a randomly selected anomalous sample. Value of the anomaly score measured by MSE for each method is specified in the figure's heading.

Reconstruction performance of the *Transformer* is significantly better than the other models. We remind that this model is not overfitted and the sample shown in this figure comes from the testing set. Both CNN and LSTM AE models for TS data need substantially longer sequence of data to be able to reconstruct well. The reason for this is that these models require enough historical context to be able to capture the underlying patterns and dependencies in the TS data. *LSTM B* AE reconstructs notably better than *LSTM A* model, but still not as good as *Transformer*.

Detected anomalies Last evaluation of models we performed is a check of match between detected sets of anomalies. Specifically, we identified the test samples with the highest anomaly scores (MSE) by selecting the last five percentiles of the scores for each model. Then, we compared the sets of samples in these percentiles between the models to determine if there were any matches. The results are displayed in detail in table 4. The greatest match is between CNN and LSTM A models, where almost 70% of their identified anomalies match.

Intersection between anomaly sets of all models is depicted in Figure 7. Each anomaly set contains 126 data samples (players) and in the intersection of the anomaly sets of all the models there are 34 players.

In the rest of the paper we restrict the discussion only on our proposed *Transformer* AE model, because it demonstrated superior reconstruction

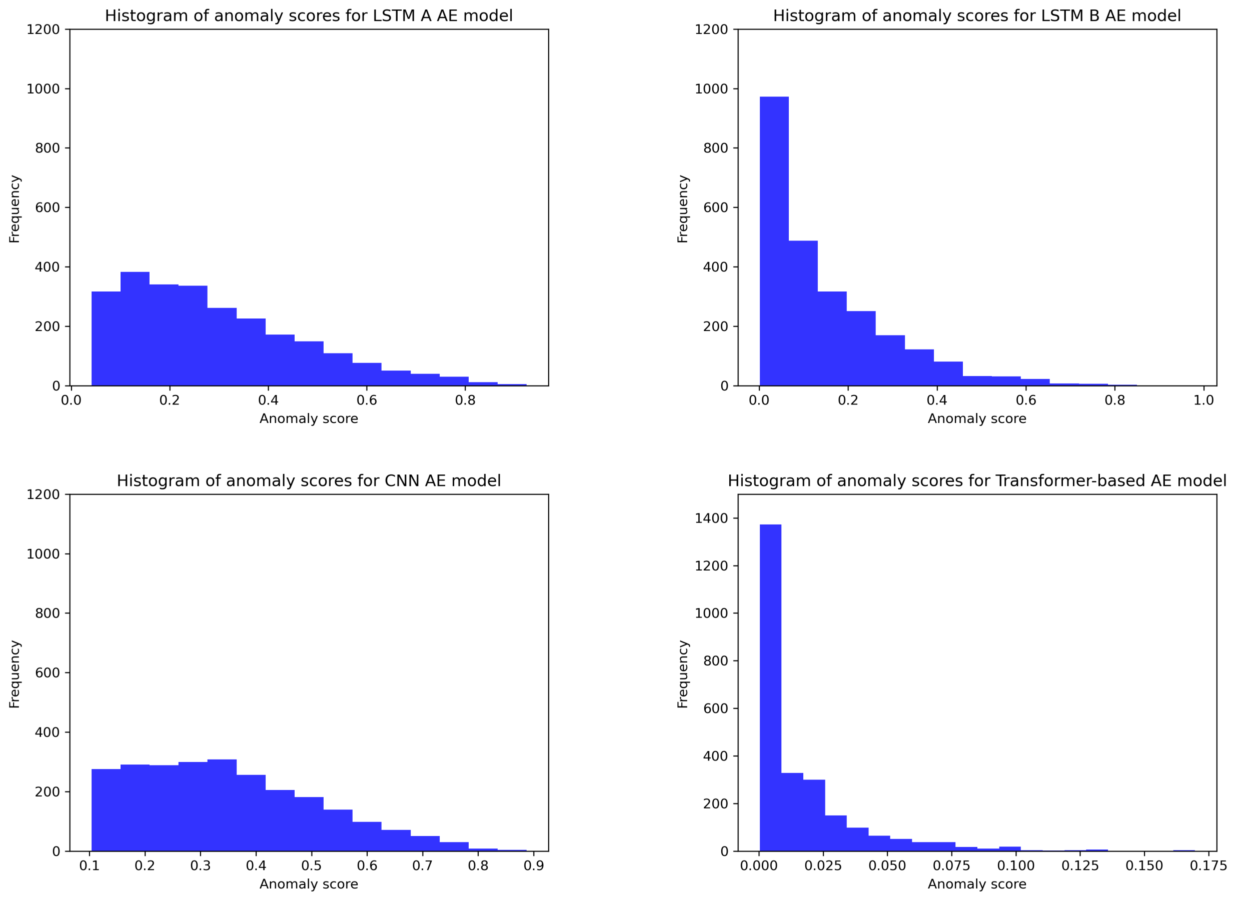


Figure 6: *Histograms of reconstruction errors for the test set for four evaluated models. The x-axis displays the value of anomaly score and the y-axis shows the frequency of the respective value.*

Table 4: *Match of anomalous test samples between evaluated AE models.*

Model name	LSTM A	LSTM B	CNN	Transformer
LSTM A	100%			
LSTM B	65.1%	100%		
CNN AE	69.8%	61.1%	100%	
Transformer	41.3%	38.9%	38.9%	100%

ability, and we thus expect, that its ability to detect anomalies is also vastly better.

Transformer-based AE model results

As discussed in the Introduction, due to the lack of clinical diagnosis in our dataset, we can only rely on proxy indicators when identifying players with potential problem gambling. Our approach is to detect the anomalies in the dataset, but we are aware that not all the anomalies must indicate the gambling disorder. Therefore, we correlated the results of the AE model with these proxy indicators:

- Mean number of logins in a time window.
- Mean number of withdrawals in a time window.
- Mean number of small and frequent withdrawals in a time window.
- Mean number of requests for the change of the deposit limit in a time window.
- Sum of the chasing episodes in the time slot of N time window.

Figure 9 depicts the correlation of the anomaly score with the proxy indicators. Each subplot

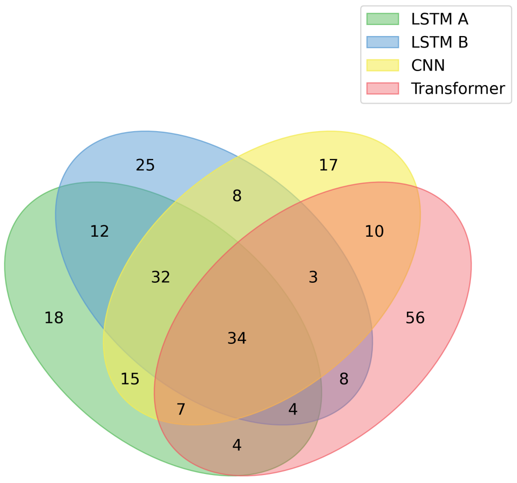


Figure 7: *Venn diagram for visualisation of intersection of detected anomaly sets of four evaluated models. The number in each part of the diagram represents the number of unique players. Anomaly set of each model contains 126 players (corresponding to 5% of testing set size).*

contains 10 bars, each bar representing one decile of the data samples (*i.e.* each bar represents 10% of data samples sorted by anomaly score). The bar colors represent the category value of the respective proxy indicator.

A distinctive pattern in players' behavior can be observed, where players with larger anomaly scores tend to exhibit high values for all the indicators evaluated. Higher frequency of logins is proportionate to higher anomaly score with more than half of the players in the last decile of reconstruction error having a mean number of logins in a time window greater than 50. The same applies for mean number of cash withdrawals in a time window. Players with low anomaly score have almost none or very few withdrawals, whilst more than one fourth of players in the last anomaly score decile have two or more withdrawals in a time window on average. Another secondary indicator we utilize is the number of small and frequent withdrawals. Most of the players with at least one of these events is in 10% of players with the highest MSE. When analyzing another indicator, namely the number of requests for a deposit limit change, we observe a more subtle pattern. It is evident that players in the first five deciles generally have no requests for a limit change (with very few exceptions), while as the anomaly score increases, the frequency of limit change requests also tends to rise. The last proxy indicator depicted is the number of chasing episodes. A rising frequency of these events proportionate to their anomaly score can be observed. More than half of the players in the last decile have at least one chasing episode in the time window.

If these plots are overlapped in order to identify the portion of players fulfilling multiple proxy indicators, following observations result: in the last five percentiles of the anomaly scores 98.6% of players satisfy at least one proxy indicator, and 77.3% satisfy at least three indicators. As for the last two percentiles, so 2% of players with the highest reconstruction error, almost 90% of them satisfy at least three indicators. The thresholds used to calculate these proportion are $>= 1$ chasing episode, $>= 1$ limit change, $>= 1$ small and frequent withdrawal, $>= 31$ logins and $>= 1.25$ withdrawal on average per time window.

Even though age and gender were not used as predictors in the models studied, we examined the demographic profile of the players in relation to the anomaly score. Figure 10 shows that there is no clear pattern in the age distribution over anomaly score deciles with every age category being present in each decile in comparable measure. As for the gender, there is a subtle pattern of women having slightly more significant presence in the first deciles with lower reconstruction error.

Conclusion

In this work, we successfully applied a transformer-based autoencoder (AE) to detect anomalies in the dataset of online casino players. The aim was to detect problem gamblers in dataset at hand in an unsupervised manner. 19 features were derived from the raw time series (TS) data reflecting players' behavior in the context of time, money and despair.

We compared the performance of this architecture with three other AE architectures based on LSTM and convolutional layers and found that the transformer-based AE achieved the best results amongst the four models in terms of recon-

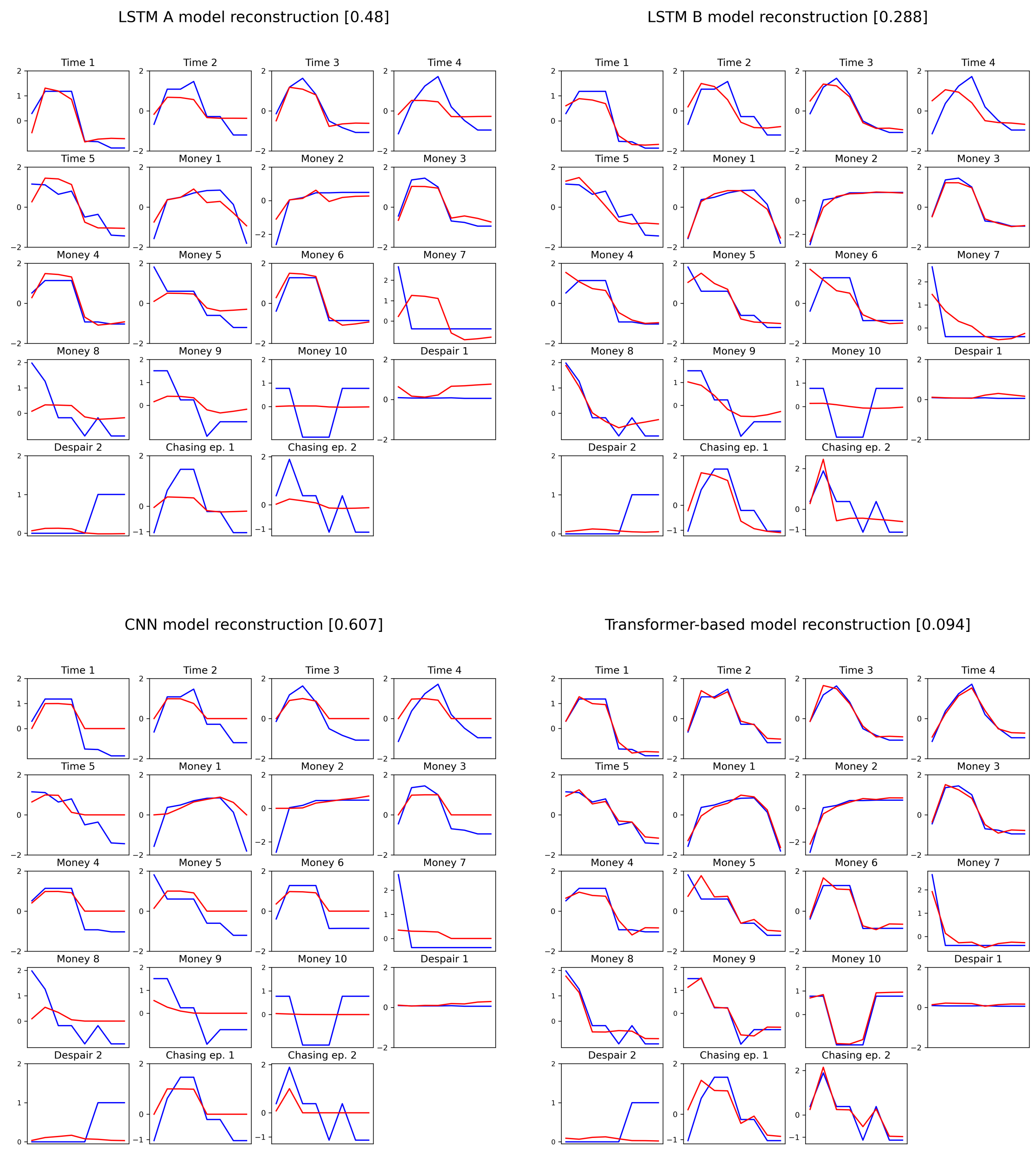


Figure 8: *Prediction performance comparison for four evaluated AE models. The same (testing dataset) data sample was reconstructed by all the models. Blue lines represent the original data and red lines represent the reconstruction by respective AE models. The number in the heading of the figures is the anomaly score of the sample.*

struction capability. This model also showcases high correlation with proxy indicators such as the number of logins, number of player's withdrawals, number of chasing episodes and other, that are commonly mentioned in literature in relation to the gambling disorder. This alignment of AE's anomaly score with proxy indicators enables us to gain insights into prediction's effectiveness in identifying players with potential problem gambling.

Even though these proxy indicators were also used as predictors, we suggest to use them as a

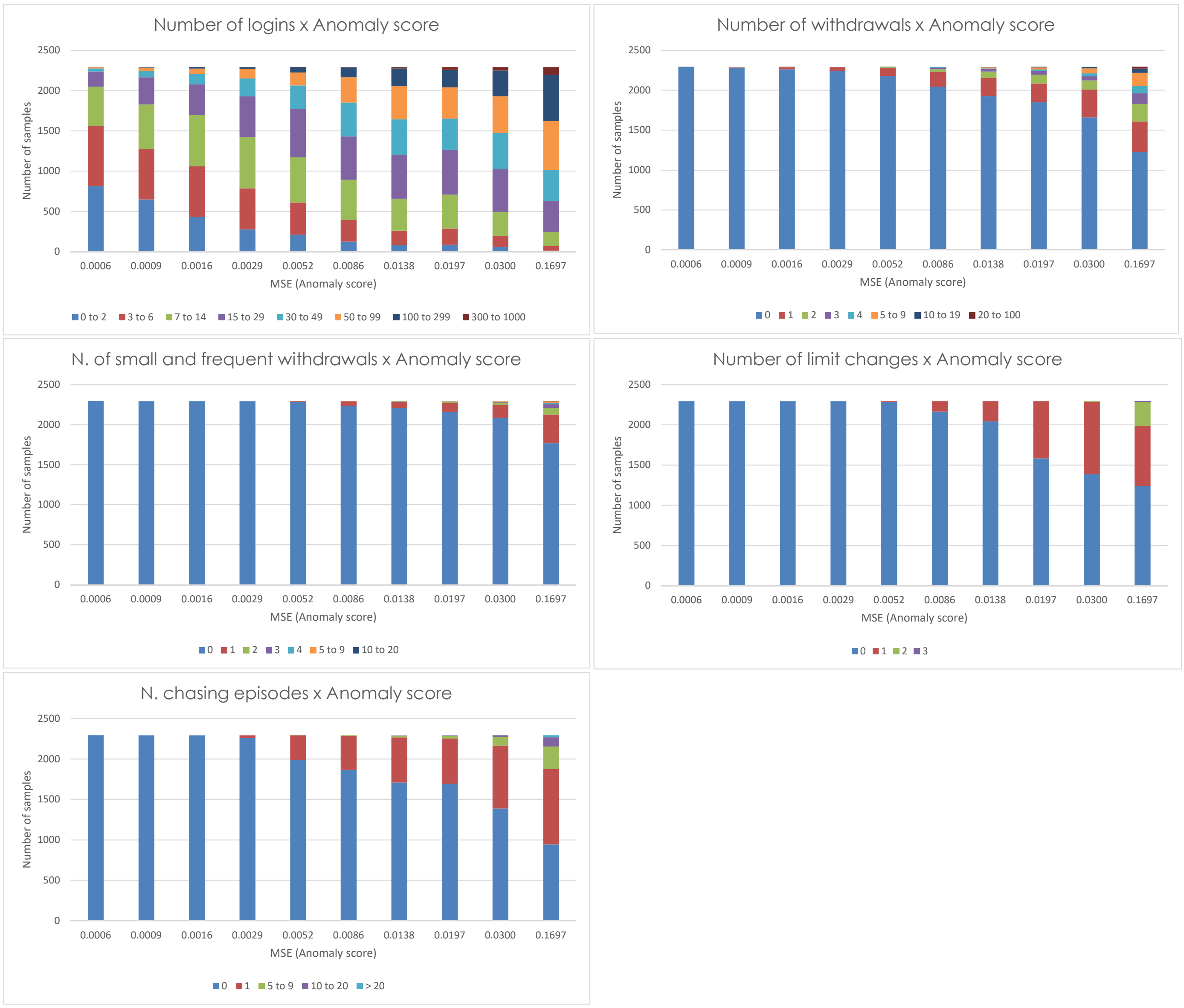


Figure 9: *Each bar in the subplot represents one decile of anomaly score (MSE). Colors represent the category of the respective proxy indicator being analyzed with category values specified in the legend.*

secondary check when detecting players with potential problem gambling in order to avoid false positives, as not all anomalies must be linked to the condition of gambling disorder.

Our findings demonstrate the potential of transformer-based AEs for unsupervised anomaly detection tasks in TS data, particularly in the context of online casino player behavior.

Acknowledgement

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the European Union's programme Digital Europe and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg , Slovakia, Norway, Switzerland, Turkey, the Republic of North Macedonia, Iceland, Montenegro and Serbia.

References

[1] Alex Blaszczynski and Lia Nower. "A Pathways Model of Problem and Pathological Gambling". In: *Addiction (Abingdon,*

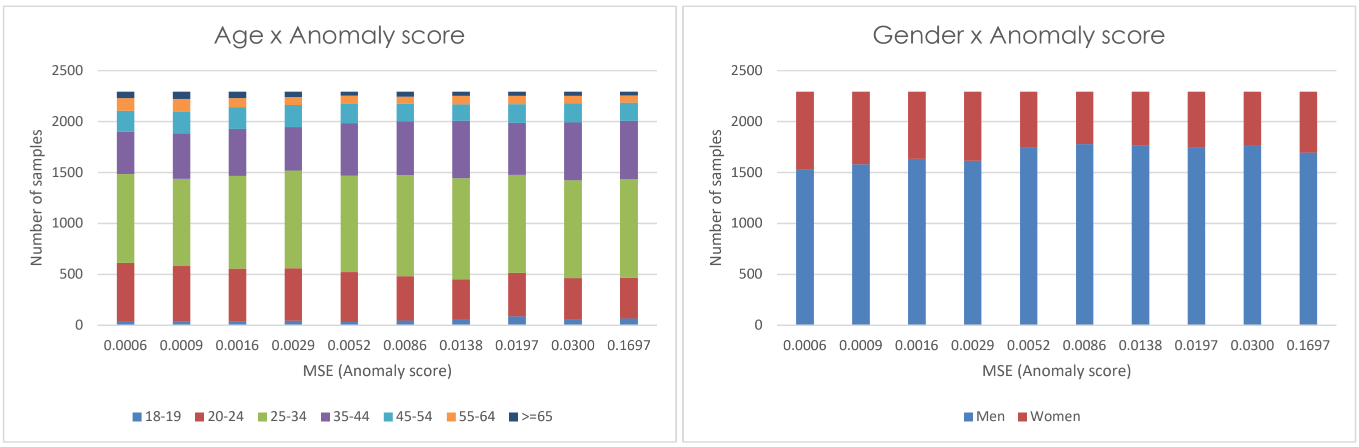


Figure 10: *Left subplot shows the relationship between age of players and anomaly score. Right subplot depicts relationship between gender of players and anomaly score. Each bar represents one decile of anomaly score (MSE), i.e. 10% of players sorted by the reconstruction error. Colors represent the age/gender category specified in the legend.*

England) 97 (June 2002), pp. 487–99. DOI: 10.1046/j.1360-0443.2002.00015.x.

[2] National Research Council. *Pathological Gambling: A Critical Review*. Washington, DC: The National Academies Press, 1999. ISBN: 978-0-309-06571-9. DOI: 10.17226/6329. URL: https://nap.nationalacademies.org/catalog/6329/pathological-gambling-a-critical-review.

[3] Luke Clark et al. "Pathological Choice: The Neuroscience of Gambling and Gambling Addiction". In: *Journal of Neuroscience* 33.45 (2013), pp. 17617–17623. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.3231-13.2013. eprint: https://www.jneurosci.org/content/33/45/17617.full.pdf. URL: https://www.jneurosci.org/content/33/45/17617.

[4] Xiaolei Deng, Tilman Lesch, and Luke Clark. "Applying Data Science to Behavioral Analysis of Online Gambling". In: *Current Addiction Reports* 6 (Sept. 2019). DOI: 10.1007/s40429-019-00269-9.

[5] Gaëlle Challet-Bouju et al. "Modeling Early Gambling Behavior Using Indicators from Online Lottery Gambling Tracking Data: Longitudinal Analysis". In: *J Med Internet Res* 22.8 (Aug. 2020), e17675. ISSN: 1438-8871. DOI: 10.2196/17675. URL: http://www.ncbi.nlm.nih.gov/pubmed/32254041.

[6] Michael Auer et al. "Development of the Online Problem Gaming Behavior Index: A New Scale Based on Actual Problem Gambling Behavior Rather Than the Consequences of it". In: *Evaluation the Health Professions* (May 2023), pp. 1–12. DOI: 10.1177/01632787231179460.

[7] Deepanshi Seth et al. "A Deep Learning Framework for Ensuring Responsible Play in Skill-based Cash Gaming". In: *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)* (2020), pp. 454–459.

[8] Leonardo Motta Perazzo Lannes. "Unsupervised Learning Applied to the Segmentation of Users of Online Gambling Platforms in Portugal". In: *NIMS - Dissertações de Mestrado em Ciência de Dados e Métodos Analíticos Avançados (Data Science and Advanced Analytics)* (Nov. 2021).

[9] Zhaomin Chen et al. "Autoencoder-based network anomaly detection". In: *2018 Wireless Telecommunications Symposium (WTS)*.

2018, pp. 1–5. DOI: 10.1109/WTS.2018.8363930.

[10] Nikolay Laptev, Saeed Amizadeh, and Ian Flint. "Generic and Scalable Framework for Automated Time-Series Anomaly Detection". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '15. Sydney, NSW, Australia: Association for Computing Machinery, 2015, pp. 1939–1947. ISBN: 9781450336642. DOI: 10.1145/2783258.2788611. URL: https://doi.org/10.1145/2783258.2788611.

[11] Saroj Gopali et al. *A Comparative Study of Detecting Anomalies in Time Series Data Using LSTM and TCN Models*. 2021. arXiv: 2112.09293 [cs.LG].

[12] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[13] Pedro Lara-Benítez et al. "Evaluation of the Transformer Architecture for Univariate Time Series Forecasting". In: *Advances in Artificial Intelligence*. Ed. by Enrique Alba et al. Cham: Springer International Publishing, 2021, pp. 106–115. ISBN: 978-3-030-85713-4.

[14] Alireza Makhzani et al. "Adversarial Autoencoders". In: *CoRR* abs/1511.05644 (2015). arXiv: 1511.05644. URL: http://arxiv.org/abs/1511.05644.

[15] Thomas Schlegl et al. "Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery". In: *CoRR* abs/1703.05921 (2017). arXiv: 1703.05921. URL: http://arxiv.org/abs/1703.05921.

[16] Xiaoran Chen and Ender Konukoglu. "Unsupervised Detection of Lesions in Brain MRI using constrained adversarial autoencoders". In: *CoRR* abs/1806.04972 (2018). arXiv: 1806.04972. URL: http://arxiv.org/abs/1806.04972.

[17] Donny Hanz. *Data Exploration with Adversarial Autoencoders*. Towards Data Science. July 2019. URL: https://towardsdatascience.com/data-exploration-with-adversarial-autoencoders-311a4e1f271b.

[18] Theodoros Ntakouris. *Time series Classification with Transformer*. 2021. URL: https://keras.io/examples/timeseries/timeseries_classification_transformer/.