A Transformer-based User Behavior Representation for Peer Grouping in Threat Detection

Xiao Lin
SMLS Group
Splunk Inc.
San Jose, CA, USA
xlin@splunk.com

Glory Avina
SMLS Group
Splunk Inc.
San Francisco, CA, USA
gloryavina@splunk.com

Stanislav Miskovic[†]
Gluware Inc.
San Jose, CA, USA
stanislav.miskovic@gmail.com

ABSTRACT

When detecting anomalies from sequence data, a machine learning model usually infers from historical data of an entity (user account or device), but ignores other peer entities' behavior, which results in false positives. Therefore, peer grouping (clustering) is often applied first and then entities are clustered into groups. This ordering allows for detection algorithms that are effective for specific peer groups to be adopted. However, performance of peer grouping is dependent on feature representation, which is dynamic for sequence data. This makes it challenging to construct good representation learning. In this study, we experiment with a method to train representation using transformer-based encoder-decoder architecture. In this architecture, we adopt sparse probability self-attention to effectively overcome transformers' limitations of large memory and a long run time. A self-attention distilling mechanism is also applied to accommodate long sequence modeling capability. Experimental results show that this proposed method is effective and applicable to solve real world problems. The trained representation is used in a downstream dynamic peer grouping task and compared against a Dynamic Time Warping (DTW) baseline. Results show that using representation learned from our system can significantly improve peer grouping performance.

CCS CONCEPTS

• Security and privacy → Intrusion/anomaly detection and malware mitigation → Intrusion detection system • Computing methodologies → Machine learning → Machine learning approaches → Learning latent representations

[†] Stanislav Miskovic was Splunk Inc staff when conducting the work related to this paper.

KDD'23 AI4Cyber, August 2023, Long Beach, CA, USA
 978-1-4503-0000-0/18/06...$15.00
https://ai4cyber-kdd.com

KEYWORDS

Anomaly detection, Representation learning, Transformer, Peer group, Self-attention

ACM Reference format:

Xiao Lin, Glory Avina and Stanislav Miskovic. 2023. A Transformer-based User Behavior Representation for Peer Grouping in Threat Detection. In *the 3rd Workshop on Artificial Intelligence-Enabled Cybersecurity Analytics* (KDD'23 AI4Cyber), Long Beach, CA, USA, 6 pages. https://ai4cyber-kdd.com.

1 Introduction

Besides matching the known bad behavior patterns, the successful detection of security threats is often manifested as a comparison between normal and anomalous behavior. Early and accurate detection of threats in real-time streaming data is imperative for most industries. Though rule intelligence-based anomaly detection is effective, especially for long tail problems, a machine learning based approach is attracting increased attention for its ability to detect novel anomaly and resilience to data variation (Pang, Shen, Cao and Hengel, 2021; Dietterich, 2002; Laptev, Amizadeh and Flint, 2015; Zhang et al., 2019; Audibert, 2020; Samtani et al., 2022). When detecting anomalies from sequence data, a machine learning model usually infers from historical data of an entity (user account or device), but ignores other peer entities' behavior, which can increase the false positive rate (Yin et al., 2020). Therefore, peer grouping (clustering) is often needed to supply the context for determining outliers. Furthermore, for internet and enterprise applications, cardinality of entities is generally too large, which prevents training/deploying a machine learning model for each single entity. Therefore, it is desirable to cluster entities into separate groups so that the entities in each group have similar behavior and then apply a suitable model to this group (Li, Zhao, Liu and Pei, 2018, Matterer and LeJeune, 2018).

To form groups for entities, a feature representation is needed for each entity. This allows an evaluation by similarity or distance. This feature representation can be derived from information about a company's hierarchy, organizational unit, or

active directory data. However, such representation suffers from two limitations: (1) it is static and not able to reflect temporal changes, or might not even be available, e.g., external attacker's account and device (Oladimeji, Ayo1 and Adewumi, 2019); (2) entities with same group identity might behave differently. For example, a user in group G_A might work in project of G_B. When a behavior model for G_A is applied on this user, many of his G_B related activities will inevitably be flagged as false positives. A user behavior representation should be built from dynamic history data instead of users' static attributes. However, it is still an open challenge to construct representation that can effectively stand for temporal dynamics of time series, especially the one proper for subsequent clustering tasks (Ma et al. 2019).

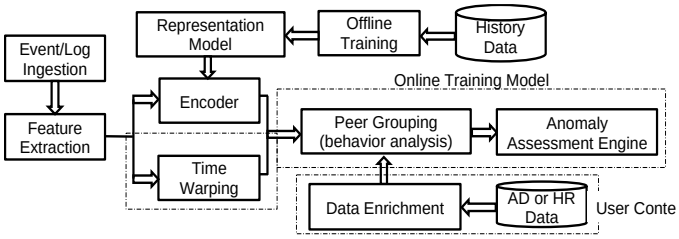

Figure 1: System Diagram of Representation Learning and Downstream Peer Grouping

In order to provide accurate representation, we experiment with a transformer-based user behavior representation for peer grouping from massive entity's historical behavior data that is readily available on Splunk's data platform. As shown in Figure 1, we have a streaming anomaly detection flow for security intelligence. Log events from data source (OS, app, or device) are ingested into the flow for feature extraction then piped to online detection flow, which includes time warping, peer grouping and anomaly assessment three modules. Our goal is to experiment with an encoder learned from offline representation learning on user history records, as an alternative to a time warping module. In this study, we construct the encoder using transformer architect and then represent it through a sequence prediction task. We successfully applied a transformer-based representation on security threat detection data. The flow trains one encoder model from multiple users' behavior history that can be used in downstream peer grouping tasks. Our end-to-end experiment shows the feasibility and effectiveness of such a system with potential application in streaming security data analytics.

2 Related Work

Success of many machine learning models largely depends on the quality of representation of input data (Bengio, Courville and Vincent, 2013, Pang, Cao, Chen, and Liu, 2018). Under the umbrella of self-supervised learning, various network architectures have been used to construct an encoder for time series representation (Li et al., 2015). Franceschi, Dieuleveut and Jaggi (2019) used causal dilated convolutions as their encoder-only-system to achieve scalable representation learning for variable length multivariate time series. Such architecture allows efficient parallelization on modern hardware like GPUs. They also first introduced triplet loss into time series unsupervised training through time-based negative sampling to enhance similarity of representation for similar time series. Because of Recurrent Neural Network (RNN)'s capability to model sequence data, it naturally gains attention in time series representation learning as well. Dezfouli et al. (2019) proposed a RNN based end-to-end learning framework to train low-dimensional representation from human decision-making behavior data. Extra terms are introduced in the loss function so that latent dimensions are informative and disentangled, i.e., encouraged to have distinct effects on behavior. In activity2vec, adversarial network is added to increase the performance and generalization of representation matrix for human body activity signals over a time segment (Aggarwal et al., 2019). Variational autoencoder (VAE) method also finds wide application in time series representation learning (Fabius and Amersfoort, 2015). In GP-VAE (Fortuin, Baranchuk, Rätsch and Mandt, 2020; Fortuin et al., 2019) VAE is used to model the low dimensional dynamics with a Gaussian process. The representation is used to map missing time series data for imputation. Pereira and Silveira (2019) also applied VAE trained with a Bi-LSTM decoder on electrocardiograms (ECG) sequence to learn representation which was used in downstream anomaly detection. Their results show that when using representation even an unsupervised algorithm can reach the accuracy level of supervised algorithm-based detection without using a latent representation.

Transformer self-attention was first proposed and used in NLP neural machine translation. Its impressive performance quickly made it one of the most popular language models and widely adopted across the NLP area. It also gained success in computer vision, and it became adopted in the time series area. Zerveas et al. (2020) for the first time applied a transformer to do representation learning on time series. They showed superior performance on several data sets against non-transformer approaches. This model is relatively small (at most hundreds of thousands of parameters), so performance was not an issue. Although attention mechanism is efficient to solve long dependency and computation parallelism, it has a well-known limitation that attention calculation has quadratic time complexity $\mathcal{O}(l^2)$ and space complexity $\mathcal{O}(J \cdot l^2)$. Many researchers have attacked this hurdle to lower the resource requirement of attention mechanism (Tay, Dehghani, Bahri and Metzler, 2020, Zhou et al., 2021).

Ma et al. (2019)'s work is like ours in the aspect that both projects target unsupervised techniques to build a representation encoder for applying clustering. Our approach differs from Ma's in two significant ways: (1) a transformer is core architecture in our encoder, while Ma's uses dilated RNN seq2seq model; (2) in Ma's system, representation learning and clustering are integrated into one flow, while in our system, they are two independent modules. The reason is that in our application, peer grouping model needs to be learned online from streaming data to facilitate

rapid response of security threat detection. Because representation learning is an independent module, the trained representation model can be adopted in other downstream machine learning tasks.

In our experiment, we used transformers to build encoders to take advantage of self-attention's power of modeling sequence dependency. The sparse query matrix approach proposed by Zhou et al. (2021) is applied to overcome the limitation of self-attention run time and memory requirement.

3 Method

3.1 Problem statement

Given a data stream emitting events $\boldsymbol{E}$(**e**, T, **x**), which reads that event $\boldsymbol{E}$ belongs to entity **e,** happens at moment t, and is featured by a vector **x**: [x_m], where $m \in M$ is dimension of feature vector, historical behavior data can be collected to form dataset $\boldsymbol{D} := \{E_t(e, T, \boldsymbol{x}),\ t \in N\}$. We then train a model $\boldsymbol{\Theta}$ offline so that for a time window $[t_0,\ t_L]$, where L is the window length, any specific entity's behavior data $\boldsymbol{E}(T_i, \mathbf{x}), i \in [0, L]$ can be encoded into a distributed representation v: [v], $\boldsymbol{\Theta}\langle \boldsymbol{E}(T_i, \mathbf{x})\rangle \rightarrow \boldsymbol{v}$. Thereafter at moment t_L the entity can be clustered online into a group G_k, $k \in K$ and K is the given number of total groups, by maximizing the similarity of entities within the same group.

3.2 Transformer-based encoder

We construct a transformer-based encoder-decoder network to learn the entity's behavior representation. The system follows the design of the original transformer architecture (Vaswani et al., 2017). A fixed length of events E'_i is fed to the encoder. The feature vector and timestamp are encoded through value and position separately. The encoder includes number (l) of identical layers to perform self-attention calculation and generates representation $\boldsymbol{v}$ after a fully connected layer. Decoder is fed with a length of L+P (L+P < N) of events E'_i that are truncated from E_i. This sequence is divided into two parts: [E'_0, E'_L] and [E'_{L+1}, E'_{L+P}]. Segment [E'_{L+1}, E'_{L+P}] is masked as zero values to train the encoder and decoder to minimize the loss function defined as mean square error (MSE) between output [y_{L+1}, y_{L+P}] and [E'_{L+1}, E'_{L+P}]. In other words, the network is trained as a sequence prediction task. After the encoder-decoder network is trained, we discard the decoder part and keep the encoder model $\boldsymbol{\Theta}$ to generate representation in the peer grouping flow.

To overcome the disadvantages of $\mathcal{O}(l^2)$ time complexity and $\mathcal{O}(J \cdot l^2)$ space complexity that exists in vanilla attention layers, we adopt the ProbSparse self-attention mechanism proposed by Zhou et al. (2021). It was shown that self-attention has long-tail distribution, i.e., there are only a few dot-products of key-query pairs that are significant, and most others can be ignored. If we approximate the i^{th} row of query's sparsity as max-mean measurement

$$\widehat{M}(q_i, K) = \max\{e^{\frac{q_i k_j^T}{\sqrt{d}}}\} - \frac{1}{L_K}\sum_{j=1}^{L_K}\frac{q_i k_j^T}{\sqrt{d}}$$

and limit query to be those with top u $\widehat{M}(q_i, K)$, where

$$u = c \cdot \ln L_Q$$

we then use a sparse query matrix $\bar{Q}$ to calculate ProbSparse self-attention as,

$$A(Q, K, V) = Softmax(\frac{\bar{Q}K^T}{\sqrt{d}})V$$

for which both time complexity and space complexity is reduced to $\mathcal{O}(L_K \ln L_Q)$.

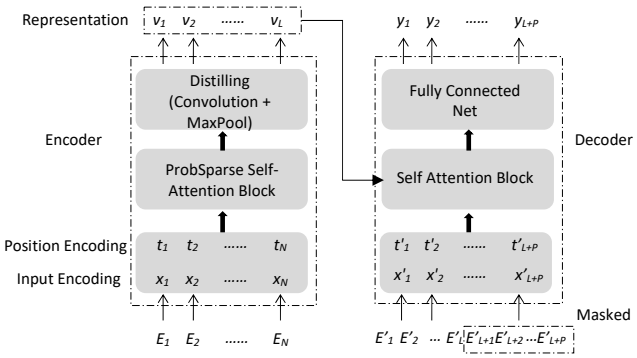

Figure 2: Diagram of Transformer-based Entity Behavior Representation Learning

As shown in Figure 2, a convolution layer plus pooling is added to distill self-attention to reduce redundant and space complexity is further reduced to $\mathcal{O}((2-\varepsilon) \cdot l \cdot \log l)$.

Unlike RNN that can use its recursive structure to capture sequential features, the transformer uses dot-product self-attention and thus relies on position encoding. For local timestamps relative to the start moment of a certain fixed time window, position embedding is simply the index of each event within the current model window. For global timestamps, we construct time features by hour frequency, yielding 4 features: hour of day, day of week, day of month and day of year. This global timestamp feature vector is added to a local timestamp position encoding to form timestamp embedding.

3.3 Online peer grouping

Our first system uses Dynamic Time Warping (DTW) as the distance measure (Petijean et al., 2014) and an online K-means (Liberty, Sriharsha and Sviridenko, 2016) to group (i.e., to cluster) peer entities based on their streaming behavior data. DTW is a widely used technique for finding similarity between two time-series but known as computationally expensive due to its pairwise similarity approach.

4 Experiments and Results

4.1 Data and setup

To form a dataset to conduct experiments in this study, we extracted data from a Window security log, which covers 23 days for a total of ~700 million events. Only these events are used: user logged on (4624), special privileges assigned (4672), Kerberos service ticket requested (4769) and account credential validation (4776). Records are aggregated with a fixed tumbling window size of 1 hour. A total of 8 features are extracted, and their distribution statistics are listed in Table 1.

	max.	min.	stdev	nonzero (%)
Authorization to Domain Controllers	47,378	45	1,564	72
Authorization to Servers	273,221	235	3,599	86
Authorization to Other Workstations	2,647,802	459	12,614	66
Distinct DCs	75	3	3	72
Distinct Servers	1,114	7	9	86
Distinct Workstations	15,221	2	73	66
Kerberos	43,554	137	2,623	86
Credential Validation	2,647,802	588	13,689	85

Table 1: Distribution Statistics of Individual Feature

All model training experiments were conducted using an AWS p3.2xlarge instance with one Nvidia V100 GPU card.

4.2 Results

We first validate whether flow is implemented appropriately by checking training convergence. As shown in Figure 3, MSE loss converges well over iteration. There is oscillation on curves with regular intervals. The explanation is that training data is switched from one entity to another in a fixed pattern because we have padded all user accounts to have the same record length.

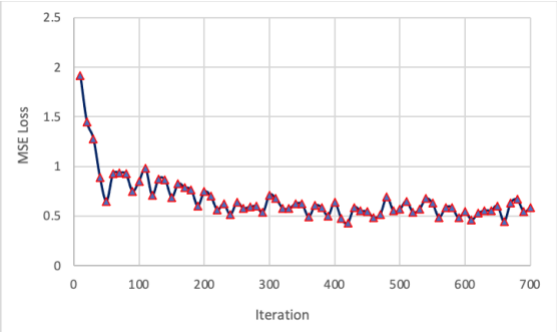

Figure 3: Training convergence: MSE loss vs. Iteration

Hyperparameter tuning is then conducted to determine a set of parameters for subsequent tests. Grid search is run to tune transformer related hyper-parameters. The search space and final selected values are listed in table below,

	Parameter Value	
	Tuning Range	Selected
Length of Input Sequence	192, 96, 48, 24	96
Number of Encoder Layers (l)	6, 4, 3, 2	3
Dimension of Hidden (d)	512, 256, 128, 64, 32	64
Head of Attention (a)	16, 8	8

Table 2: Hyperparameters of Transformer Encoder

The network with chosen hyper-parameters has about 11 million trainable parameters. Using this optimized configuration, we show that the sequence can be successfully constructed, as visually shown in Figure 4 and quantitatively by the reconstruction error (MSE).

$$\epsilon_{MSE} = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{L \times P}\sum_{i=1}^{L}\sum_{j=1}^{P}\sum_{m=1}^{M}\left(\tilde{x}_{i,j}[t,m] - x_{i,j}[t,m]\right)^2$$

where N is number of test sequence, L is length of context sequence, P is the length of marked data point (prediction length in sequence prediction task), and M is the dimension of feature vector. The average is 0.0985 in range of [0.0411, 0.2399] across our hyperparameter grid search results.

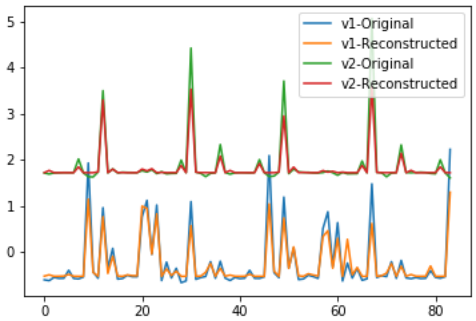

Figure 4: Original and reconstructed data for sampled 2 features

We validate computation resource consumption of probability self-attention, and the result is shown in Figure 5. It roughly shows $\mathcal{O}(L \ln L)$ trend for both runtime and GPU memory consumption, which is the expected behavior.

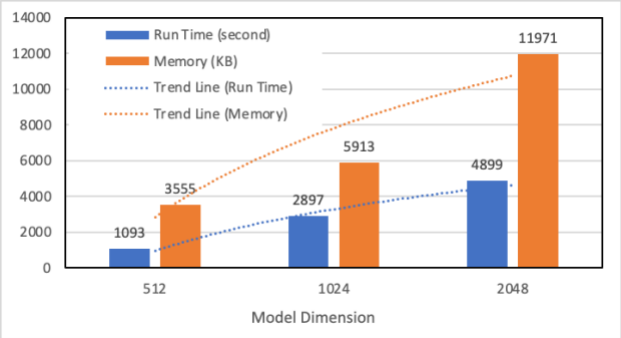

Figure 5: Run time and memory efficiency of probability self-attention mechanism.

We now visualize the clusters formed by the latent representation. The representation encoder trained from historical data was applied on the newly streamed-in data from each entity and generated corresponding representation vector. We then do PCA and use t-SNE to generate a 2D view of these vector's distribution as shown in Figure 6. As we can see from the plot, representation vectors form several clusters with good concentrations. This shows that our encoder can catch the behavior characteristics from devices' activity records and form the right representation. It should be noted that in our application, peer grouping is used as a data preparation step before the threat assessment engine. Therefore, the count of clusters is not essential. Instead, separation of clusters and their density is more important for subsequent tasks to achieve better performance.

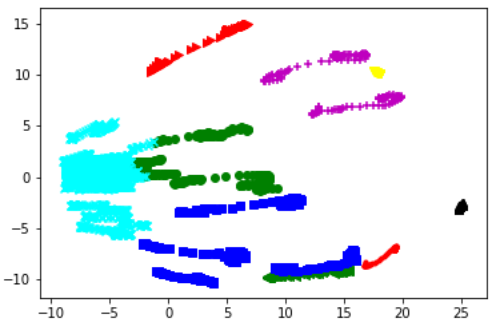

Figure 6: Distribution of representation of vectors generated from transformer-based encoder

Finally, we compare the trained representation vectors with time warping to validate that our new encoder approach can generate better input for the downstream peer grouping module. We use the Davies-Bouldin Index (Halkidi, Batistakis and Vazirgiannis, 2001) to measure the quality of representation generated from the transformer-based encoder and time warping. Considering the infinite length of streaming data, we use data in a certain length to evaluate the index. Various time windows sizes are chosen to perform comparison. Results are summarized in Table 3. As shown in the table, our encoder approach outperforms time warping approach for various time window sizes. Also, it can be noticed that time warping's performance declines as the window size increases, while our encoder approach stays relatively stable. It is persuasive that this encoder approach generates a better separation between groups that downstream peer grouping modules will benefit from.

Window Size (hours)	Davies-Bouldin Index (K=10)	
	Time Warping	Encoder
8	0.8178	0.7586
24	1.0555	0.5770
48	1.2213	0.7171

Table 3: Cluster performance comparison between representation from transformer-based encoder and time warping

SUMMARY

We proposed and showed a flow to use a transformer-based encoder to learn representation of entity behavior from historical data. Transformer's self-attention is calculated based on dot-product probability distribution to generate sparse query matrix, so that both time and space complexity can be reduced to $\mathcal{O}(L \ln L)$. Convolution and max pooling layers are added to further reduce the resource requirement of the attention mechanism. The learned representation is successfully applied to a subsequent peer grouping task on real-world security log data. Results show that representation obtained in this flow can improve peer grouping's performance. Potentially, other downstream machine learning tasks for threat detection might benefit from this representation as well.

As next steps, we want to improve the scalability of the proposed method and explore a more advanced architecture. It will also be interesting to conduct benchmarks against other newly published representation learning techniques such as temporal neighborhood coding (TNC) (Tonekaboni, Eytan and Goldengerg, 2021).

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments to improve this work.

REFERENCES

[1] Kara Aggarwal et al., 2019, Adversarial Unsupervised Representation Learning for Activity Time-Series, in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (AAAI-19), pp. 834-841.

[2] Julien Audibert, 2020, USAD: UnSupervised Anomaly Detection on Multivariate Time Series, in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Virtual, USA, https://doi.org/10.1145/3394486.3403392.

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent, 2013, Representation Learning: A Review and New Perspectives, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798-1823, https://doi.org/10.1109/TPAMI.2013.50.

[4] Amir Dezfouli et al., 2019, Disentangled behavioral representations, in *Proceedings of the 33rd Conference on Neural Information Processing Systems* (NeurIPS 2019), Vancouver, Canada

[5] Thomas G. Dietterich, 2002, Machine Learning for Sequential Data: A Review, in *Proceedings of Joint SSPR/SPR, Structural, Syntactic, and Statistical Pattern Recognition*, pp. 15-29.

[6] Otto Fabius and Joost R. van Amersfoort, 2015, Variational Recurrent Auto-encoders, in *International Conference on Learning Representation Workshop*, San Diego, USA.

[7] Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch and Stephan Mandt, 2020, GP-VAE: Deep Probabilistic Time Series Imputation, in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, PMLR 108:1651-1661.

[8] Jean-Yves Franceschi, Aymeric Dieuleveut and Martin Jaggi, 2019, Unsupervised Scalable Representation Learning for Multivariate Time Series, in *Proceedings of the 33rd Conference on Neural Information Processing Systems* (NeurIPS'19), Vancouver, Canada.

[9] Vincent Fortuin, Matthias Hüser et al., 2019, SOM-VAE: Interpretable Discrete Representation Learning on Time Series, in *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA.

[10] Maria Halkidi, Yannis Batistakis and Michalis Vazirgiannis, 2001, On Clustering Validation Techniques, *Journal of Intelligent Information Systems*, vol. 17, pp. 107–145, https://doi.org/10.1023/A:1012801612483.

[11] Nikolay Laptev, Saeed Amizadeh and Ian Flint, 2015, Generic and Scalable Framework for Automated Time-series Anomaly Detection, in *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1939-1947, https://doi.org/10.1145/2783258.2788611.

[12] Zhihan Li, Youjian Zhao, Rong Liu and Dan Pei, 2018, Robust and Rapid Clustering of KPIs for Large-Scale Anomaly Detection, in *Proceedings of IEEE/ACM 26th International Symposium on Quality of Service* (IWQoS'18), https://doi.org/10.1109/IWQoS.2018.8624168.

[13] Jundong Li et al., 2015, Unsupervised Streaming Feature Selection in Social Media, in *Proceedings of the 24th ACM International Conference on Information and Knowledge Management* (CIKM'15), pp. 1041–1050, https://doi.org/10.1145/2806416.2806501.

[14] Edo Liberty, Ram Sriharsha and Maxim Sviridenko, 2016, An Algorithm for Online K-Means Clustering, in *Proceedings of the 18th Workshop on Algorithm Engineering and Experiments* (ALENEX'16), pp. 81-89, https://doi.org/10.1137/1.9781611974317.7

[15] Qianli Ma et al., 2019, Learning Representations for Time Series Clustering, in *Proceedings of the 33rd Conference on Neural Information Processing Systems* (NeurIPS 2019), Vancouver, Canada.

[16] Jason Matterer and Daniel LeJeune, 2018, Peer Group Metadata-Informed LSTM Ensembles for Insider Detection, in *Proceedings of the 31st International Florida Artificial Intelligence Research Conference* (FLAIRS-31), pp. 62-67.

[17] Tolulope O. Oladimeji, C. K Ayo1 and S.E Adewumi, 2019, Review on Insider Threat Detection Techniques, *Journal of Physics: Conf. Series* 1299, https://doi.org/10.1088/1742-6596/1299/1/012046.

[18] Guansong Pang, Longbing Cao, Ling Chen and Huan Liu, 2018, Learning Representations of Ultrahigh-dimensional Data for Random Distance-based Outlier Detection, in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2041-2050, https://doi.org/10.1145/3219819.3220042.

[19] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel, 2021, Deep Learning for Anomaly Detection: A Review, *ACM Computing Survey*, March 2021, article no.: 38, https://doi.org/10.1145/3439950.

[20] João Pereira and Margarida Silveira, 2019, Unsupervised Representation Learning and Anomaly Detection in ECG Sequences, *International Journal of Data Mining and Bioinformatics*, vol. 22, no. 4, pp. 389-407.

[21] Francois Petijean et al., 2014, Dynamic Time Warping Averaging of Time Series allows Faster and more Accurate Classification, in *Proceedings of 2014 IEEE International Conference on Data Mining*, pp. 470-479, https://doi.org/10.1109/ICDM.2014.27.

[22] Sagar Samtani, Gang Wang, Ali Ahmadzadeh, Arridhana Ciptadi, Shanchieh Yang, Hsinchun Chen, 2022, ACM KDD AI4Cyber/MLHat: Workshop on AI-enabled Cybersecurity Analytics and Deployable Defense, *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, August 2022, pp. 4900–4901, https://doi.org/10.1145/3534678.3542894.

[23] Yi Tay, Mostafa Dehghani, Dara Bahri and Donald Metzler, 2020, Efficient Transformers: A Survey, arXiv:2009.06732v2.

[24] Sana Tonekaboni, Danny Eytan and Anna Goldengerg, 2021, Unsupervised Representation Learning for Time Series with Temporal Neighborhood Coding, in *Proceedings of the 9th International Conference on Learning Representations* (ICLR'21), Virtual Conference.

[25] Ashish Vaswani et al., 2017, Attention Is All You Need, in *Proceedings of the 31st Conference on Neural Information Processing Systems* (NIPS 2017), Long Beach, CA, USA.

[26] Jianwen Yin et al., 2020, Learning Transferrable Parameters for Long-tailed Sequential User Behavior Modeling, in *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (KDD'20), August 23-27, 2020, Virtual Conference, https://doi.org/10.1145/3394486.3403078.

[27] Chuxu Zhang et al., 2019, A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data, in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (AAAI-19), pp. 1409-1416, Hawaii, USA.

[28] George Zerveas et al., 2020, A Transformer-based Framework for Multivariate Time Series Representation Learning. arXiv preprint arXiv:2010.02803v3.

[29] Haoyi Zhou et al., 2021, Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting, in *Proceedings of the 35th AAAI Conference on Artificial Intelligence* (AAAI-21), Virtual Conference, Canada.