UDC 004.75
DOI: 10.15587/1729-4061.2024.316779

DEVELOPMENT A SET OF MATHEMATICAL MODELS FOR ANOMALY DETECTION IN HIGH-LOAD COMPLEX COMPUTER SYSTEMS

Yelyzaveta Meleshko
Doctor of Technical Sciences, Professor
Department of Cybersecurity and Software*
Mykola Yakymenko
PhD, Associate Professor
Department of Higher Mathematics and Physics*
Volodymyr Mikhav
Doctor of Philosophy in Computer Engineering
Department of Information Technologies**
Yaroslav Shulika
PhD Student
Department of Cybersecurity and Software*
Viacheslav Davydov
Corresponding author
Doctor of Technical Sciences, Associate Professor
Department of Information Technology and Cyber Security**
E-mail: vyacheslav.v.davydov@gmail.com
*Central Ukrainian National Technical University
Universitetsky ave., 8, Kropyvnytskyi, Ukraine, 25006
**Science Entrepreneurship Technology University
Mykoly Shpaka str., 3, Kyiv, Ukraine, 03113

The subject of this study is the process of anomaly detection in high-load complex computer systems (HLCCSs). The task addressed in the paper is the lack of real-time anomaly detection models in HLCCS with a specified accuracy. A set of mathematical models for real-time anomaly detection has been built and investigated. This set includes a mathematical model for detecting anomalous connections between components of computer system (DACCCSs) and a mathematical model for assessing current state of computer system (CSACS).

The results of models tests showed the following efficiency metrics. For a DACCCS model: accuracy – 84 %, positive predictive value – 87 %, recall – 74 %, and weighted average accuracy (WAA) – 78 %. For a CSACS model: accuracy – 91 %, positive predictive value – 82 %, recall – 68 %, and WAA – 67 %.

The positive results of the study can be attributed to the following factors. A DACCCS model uses projection matrices and orthogonal vector functions to analyze anomalies. This enables the creation of spatial decompositions that reveal complex interrelationships between system components using only eigenvalues and eigenvectors. A CSACS model applies the singular value decomposition method, which implies solving a system of scalar equations to determine the current state of the system. This approach minimizes computational costs compared to methods requiring the solution of complex matrix equations. Thus, the model could be applied for real-time data analysis and anomaly detection under conditions of limited resources and high system load.

The practical application scope includes HLCCS, such as banking transaction servers and cloud platforms, in which it is essential to enable stable operation under high request amount and to minimize the risk of data loss or service failure

Keywords: high-load complex computer systems, anomaly detection, mathematical models, real-time

Received 05.09.2024
Received in revised form 06.11.2024
Accepted 25.11.2024
Published 30.12.2024

How to Cite: *Meleshko, Y., Yakymenko, M., Mikhav, V., Shulika, Y., Davydov, V. (2024). Development a set of mathematical models for anomaly detection in high-load complex computer systems. Eastern-European Journal of Enterprise Technologies, 6 (4 (132)), 14–25. https://doi.org/10.15587/1729-4061.2024.316779*

1. Introduction

High-load web services and computer systems are a key part of modern infrastructure as they enable the stable operation of banking services, commercial platforms, online educational resources, social networks, and many other critical industries. Given the significant load that these systems withstand on a daily basis, it is necessary to comply with high requirements for security and reliability indicators for their stable operation.

One of the main challenges for such systems is detecting anomalies in real time. Anomalies can signal system malfunctions, process inconsistencies, or potential cyber attacks. Highly loaded systems are particularly sensitive to even minor failures as they can cause significant delays or total unavailability of service for many users at the same time, resulting in financial losses and loss of customer trust.

Modern web services face such challenges as DDoS attack attempts, significant drops in user requests, database connectivity issues, memory leaks, and the impact of unexpected changes in network or hardware configuration. Therefore, ensuring the timely detection of anomalies is a critically important aspect of the security of highly loaded systems. Automation of the analysis process and timely identification of potential problems in real time make it possible to reduce risks and minimize downtime.

Highly loaded systems are characterized by complex process dynamics and a large number of components, which requires the use of tools for deep analysis of the interaction between them. One of the promising approaches is the use of models for detecting anomalous connections between system components, which makes it possible to evaluate the stability of behavior, taking into account complex dependences between the load on the processor, memory, and network components.

Considering these requirements, the development of a mathematical model for detecting anomalies in highly loaded web services aims not only to increase the detection

accuracy but also to ensure a timely response to potential threats. This makes it possible to avoid significant financial losses and increase the level of trust of users in such services.

Therefore, it is a relevant task to build mathematical models for the detection of anomalies in highly loaded complex computer systems, which have the capability to obtain results in real time with a predefined accuracy.

2. Literature review and problem statement

Study [1] analyzed the use of in-memory computing technologies to speed up the operation of highly loaded deep learning systems. Special emphasis is on the prospects of improving energy efficiency and reducing delays when processing large volumes of data, which is especially relevant for scalable computer systems that serve numerous requests at the same time. The authors provide an overview of modern trends and forecasts of technology development. In the conclusions, the authors emphasize the relevance of research into highly loaded complex computer systems in various directions, including security and reliability. However, given the generalized nature of the work, the authors do not provide results of practical experiments and do not specify the achieved advantages of the research.

Paper [2] describes modern trends in the development of network intrusion detection systems, which are critically important for the protection of highly loaded complex computer systems from cyber threats. The authors describe in detail the current state of methods for detecting anomalies in network flows and discuss the need to create adaptive and scalable real-time solutions that will help ensure the reliable operation of such systems even during peak loads. However, there are still unresolved issues regarding the coordination of different methods of anomaly detection and the adaptation of these methods to new threats. The main reason for this is the lack of a single standard for evaluating and comparing the performance of different models, as well as the high demands on computing resources required to implement these approaches in real time.

Paper [3] describes the main interconnection networks used in supercomputing systems, such as NVIDIA InfiniBand, Intel Omni-Path, and Cray Slingshot, and presents their evolution and development trends under conditions of decreasing Moore's law. Special attention is paid to the performance of networks, as well as the speed of reaction to possible impacts. The authors emphasize that these characteristics are key to increasing the efficiency of parallel computing in high-performance computers. This is important for understanding future challenges in the development of network solutions for highly loaded systems. However, several unsolved issues remain, such as ensuring the scalability of these networks to further improve performance. This is related to issues of technical limitation, architectural features of modern network solutions and requirements for computing resources.

Our review of the literature [4–8] showed that a number of scientific works report the mathematical modeling of processes in highly loaded systems with an emphasis on the possibility of data protection. So, for example, in paper [4], the authors propose a mathematical model for detecting anomalies in complex computer systems. The authors emphasize the need for a universal and scientifically based approach to monitoring the behavior of a highly loaded system. The research aims to establish a general criterion for identifying anomalous activity based on the homogeneity of input data samples. By improving the sampling homogeneity criteria and isolating observations indicating anomalous behavior, the proposed model aims to improve the reliability and security of computer systems. However, the model is designed only for processing series from small samples. In addition, the model has a high computational complexity.

Paper [5] offers methods for probabilistic analysis and modeling of the dynamics of complex systems, which can be used for modeling highly loaded complex computer systems and analyzing their reliability and behavior under conditions of abnormal changes. This meets the goals of building models to detect anomalies in the behavior of systems, especially with an emphasis on modeling and understanding the dynamics of transitions between states. The authors proposed a mathematical model for calculating the conditional probabilities that the system would be in a certain state at a specific time, provided that the system was in any of the possible states at the initial moment. But the limitation of the provided graphical modeling approach, which assumes the presence of only three states, reduces the practical value of the proposed model under the conditions of complication of the security situation in real time.

In paper [6], mathematical modeling of the process of system security testing was carried out using a proven graph modeling method with the addition of elements from the theory of fuzzy logic. That made it possible to mathematically formalize a complex process and obtain mathematical expressions for calculating probability-time characteristics. A similar approach is reported in [7]. But the authors of the models did not verify them on highly loaded complex computer systems. Therefore, it is possible to draw conclusions about the expediency of other approaches to the mathematical formalization of complex technical systems.

One of these approaches is described in paper [8]. In it, the authors recommend mathematically formalizing anomaly detection systems using the basic principles of dynamic chaos theory. In particular, the authors suggest using the BDS statistic (Brock-Dechert-Sheinkman statistic) as the main indicator. This is a modern mathematical apparatus for detecting anomalies in time series, which can be used when describing the indicators of information processes. In general, the mathematical apparatus from the theory of dynamic chaos has prospects in the formalization of particularly complex highly loaded complex computer systems. However, the authors of the paper did not analyze the complexity and speed of the anomaly detection process.

The authors of [9] consider the development of computing paradigms, in particular cloud, edge, and fog computing, in the context of highly loaded systems. It focuses on transforming cloud technologies into more distributed approaches, such as Edge Computing, to improve data processing efficiency, reduce latency, and optimize resource management. The study discusses the current challenges and prospects for implementing machine learning to improve system performance. The paper highlights the need to use ML (Machine Learning) to optimize speed, accuracy, security, and low power consumption. This is important for highly loaded systems that must ensure minimal delays and stable operation under conditions of large amounts of data. At the same time, the authors do not provide quantitative indicators of speed.

The authors of paper [10] propose an approach to dynamic load balancing in cloud computing that uses deep learning. The authors explore a method that combines neural networks and reinforcement learning with hybrid optimization algorithms to improve the efficiency of load balancing and

task management. The study confirms the importance of reducing latency, optimizing CPU (Central Processing Unit), and memory usage to improve the performance of high-load systems. This emphasizes the importance of these indicators when detecting anomalous system behavior. However, several unsolved issues remain, such as ensuring the scalability of these approaches for large cloud systems and accounting for unpredictable changes in load. The main reason for this is the complexity of dynamic cloud environments, in which it is necessary to adapt to constantly changing conditions and various load characteristics.

Another example of the use of artificial intelligence in modeling complex computer systems is [11]. The paper discusses the optimization of the large BERT (Bidirectional Encoder Representations from Transformers) language model to run on multi-core CPUs. The study shows how porting and parallelizing BERT to a multi-core ARM (Advanced RISC Machine) processor can significantly reduce model training time, in particular through the use of multithreading. The paper investigates the effect of parameters such as batch size and learning rate on model performance in various natural language processing tasks. But it does not show the possibility of using the model in real time.

Paper [12] discusses the use of machine learning technologies to improve the efficiency of resource management in heterogeneous cloud computing systems. It describes approaches to the allocation of resources and optimization of calculations, which is related to the problems of highly loaded complex computer systems. Optimizing resource management, especially in cloud systems, is an important part of ensuring the reliability and efficiency of such systems. However, the authors of the paper focused on certain reliability indicators without taking into account possible cybernetic influences on the system from the outside. And this is one of the requests for the future.

Based on our review of the literature [1–12], several critical unresolved issues that require further research were identified. First of all, one of the key challenges is to increase the speed of anomaly detection, taking into account the accuracy requirements for HLCCSs. This is especially relevant under the conditions of performance of tasks by such systems in real time. Also, the issues of scalability of anomaly detection methods, which must work effectively with a large amount of data and change according to the dynamic conditions of the environment, remain unsolved. The lack of a single standard for evaluating the effectiveness of different approaches makes it difficult to objectively compare the results, which creates barriers to the introduction of new methods in industrial applications. But it can be noted that the processes of scalability of new methods depend to a large extent on the speed of task execution.

Another important issue is the adaptation of existing models to dynamic conditions, which can significantly affect their accuracy and speed of calculations. Further research should focus on the development of more flexible models that can quickly adapt to environmental changes, which will reduce delays in the system's response to potential threats.

3. The aim and objectives of the study

The purpose of our work is to build a set of mathematical models for detecting anomalies in highly loaded complex computer systems based on the provisions from the theory of dynamic chaos. This will make it possible to increase the speed of detection of anomalies in the behavior of highly loaded complex computer systems, which, in turn, should increase their security.

To achieve the goal, the following tasks were set:

– to investigate a generalized model for determining the stochastic sensitivity of cycles for highly loaded web services and complex computer systems;

– to build and research a mathematical model for detecting anomalous connections between computer system components;

– to develop and examine a mathematical model for assessing the state of the computer system at a current time point or at short time intervals.

4. The study materials and methods

The object of our study is the process of detecting anomalies in HLCCS. The subject of research is a set of mathematical models for detecting anomalies in real time.

The application of a comprehensive approach, which includes mathematical models built on the basis of the theory of dynamic chaos, as well as machine learning methods, will make it possible to significantly increase the speed of detecting anomalies while ensuring the necessary accuracy, which is critically important for the effective operation of real-time systems.

It is assumed that computer systems have a sufficient number of sensors to collect data about the state of components. It is also assumed that abnormal changes in the behavior of the system can be detected based on the analysis of deviations in the interaction between components.

To simplify the simulation, it is assumed that all anomalies have time-independent characteristics, and the system operates under conditions of constant average load, and short-term peaks have little effect on the stability of the system.

Mathematical models were used to detect anomalies in highly loaded complex computer systems, which make it possible to analyze system dynamics, assess sensitivity, and detect possible deviations in system behavior. The theoretical basis of the research is the use of fundamental principles from the theory of dynamic chaos to model sensitivity, as well as the use of statistical methods to identify anomalies in the behavior of the system.

Python software (version 3.9) with a set of libraries was used for simulation and analysis: Scikit-Learn for machine learning and data processing, TensorFlow for creating neural networks, Pandas for table processing, and Matplotlib for visualizing results. Libraries for solving differential equations (SciPy) were also used.

The experiment was conducted under conditions of high load on the servers, with the simulation of various scenarios of network activity and processor operation. Data from the public dataset CIC-IDS-2018, which contains information on normal and abnormal network activity, was used. The system worked in real time, which allowed us to evaluate its performance and ability to detect anomalies.

The following configuration was used during the experiment. The network was built according to a star topology, where a central server acted as a coordinator and provided communication with several client nodes. A high-performance server based on an Intel Core i5-10400 processor was used. The server functioned as a coordinator for traffic management and request processing. 10 client nodes were connected, each

of which was a PC with an ARM architecture processor. Each of the nodes carried out a simulation of the user load, which included the generation of requests to the central server. The TP-Link TL-SG108E router was used, which ensured reliable distribution of traffic between clients. Managed switches with QoS (Quality of Service) support were used to connect all client nodes. All connections were made using Category 6 Ethernet cables (Cat6), which provided a data transfer rate of up to 1 Gbit/s. The router was connected to the main server and provided access to the Internet, which allowed testing under conditions of mixed traffic (local and external).

During the simulation, it was assumed that all client nodes have the same performance and load, which allowed us to evaluate general trends in the system's operation. It was also assumed that there are no significant delays in the network and that the network connections are stable throughout the duration of the experiment.

After obtaining the initial data, their pre-processing, including standardization and normalization, was carried out. The data were divided into training and test samples in the ratio of 70:30. To ensure the balance of the classes, the SMOTE (Synthetic Minority Over-sampling Technique) algorithm was used, which makes it possible to increase the number of examples for the less represented class.

To check the adequacy of the models built, cross-validation methods were used, as well as accuracy metrics, such as accuracy, completeness (recall), prediction accuracy (precision), and F1-measure. A comparative analysis of models with different approaches to anomaly detection (Isolation Forest, Autoencoder, One-Class SVM) was also conducted.

5. A set of mathematical models for detecting anomalies in a computer system based on the sensitivity of 3D cycles

5. 1. A generalized model for determining the stochastic sensitivity of cycles for highly loaded web services and complex computer systems

We consider the case when the invariant multi-species M of the system is a limit cycle. Such a cycle can be given by some τ-periodic solution $x=\xi(t)$, where $x_0=\xi(0)$ is a fixed point of the cycle. The solution on the interval $[0, \tau)$ specifies the natural parameterization of the cycle points: $M=\{\xi(t)|0\le t\le\tau\}$. It is assumed that the cycle M is E-stable. In this case, a stationary distributed bundle of random trajectories of a complex system is formed around the cycle. At the same time, analysis of the stochastic sensitivity of the multispecies M according to the general theory for a complex system is reduced to the construction and study of the τ-periodic solution $W(t)$ of the matrix equation:

$$V = F(t)V + VF^T(t) + P(t)S(t)P(t), \quad (1)$$

with T-periodic coefficients:

$$F(t) = \partial f / \partial x(\xi(t)), S(t) = G(t)G^T(t),$$

$$G(t) = \sigma(\xi(t)), P(t) = P_{\xi(t)}.$$

The matrix $W(t)$ – the stochastic sensitivity function of the cycle M – is the only solution to equation (1) in the space Σ of symmetric $n\times n$ matrices defined and sufficiently smooth on R^1 with periodicity conditions:

$$\forall t \in R^1 : V(t+\tau) = V(t), \quad (2)$$

and innateness:

$$\forall t \in R^1 : V(t)r(t) = 0,\ r(t) = f(\xi(t)). \quad (3)$$

The matrix $W(t)$ here has the following probabilistic interpretation.

A stochastic system is considered:

$$dy = F(t)ydt + P(t)G(t)dw(t). \quad (4)$$

This system has a certain periodic regime associated with the solution $\bar{y}(t)$. Covariance matrix of a periodic random process $\bar{y}(t)$ is the desired solution $W(t)$ to system (1) to (4), which corresponds to the stable state of the system. It is the result of the asymptotic behavior of the process and provides a characteristic of the connections between the components of the system in the long term.

Random trajectories of a linear complex system are formed around a cycle of bundles that lie in a certain invariant neighborhood for a complex system of domain U. Let PL_t be a hyperplane orthogonal to the cycle at the point $\xi(t)$ $(0\le t\le Q)$. The neighborhood of the point $\xi(t)$ lying in PL_T is denoted by U_t: $U_t=U\cap PL_t$. It is assumed that $U_t\cap U_s=\varnothing$ if $t\ne s$.

It is convenient to associate the probabilistic description of random trajectories in the beam with the vector function X_t. The X_t values are the intersection points of the random trajectories of the linear complex system in U_t. The probability distribution of trajectories in the beam stabilizes over time, so the random variable X_t in the neighborhood U_t has a certain stationary distribution with density $\rho_t(x,\varepsilon)$.

The noise function $\rho_t(x,\varepsilon)$ near the cycle has exponential Gaussian asymptotic $\rho_t^*(x,\varepsilon)$:

$$\rho_t(x,\varepsilon) \approx \rho_t^*(x,\varepsilon) K_{\exp}\left(\frac{-(x-\xi(t))^T W^+(t)(x-\xi(t))}{2\varepsilon^2}\right),$$

with the mean value $m_t=\xi(t)$ and the covariance matrix $m_t=\xi(t)$ given by the stochastic sensitivity function $W(t)$.

This distribution, concentrated in the PL_t hyperplane, is singular $\operatorname{rank} D(t,\varepsilon)\le n-1$. For non-degenerate noises $(\det\sigma(x)|M\ne 0)$ $\operatorname{rank} D(t,\varepsilon)\le n-1$ is considered. The covariance matrix $D(t,\varepsilon)$ characterizes the spread of intersection points of random trajectories with the hyperplane PL_t.

The eigenvalues $\lambda_1\ge\lambda_2\ge\ldots\ge\lambda_n\ge 0$ and the eigenvectors $v_1, v_2, \ldots, v_n$ of the matrix $W(t)$ are considered. Due to the degeneracy of $W(t)$, the eigenvalue is $\lambda_n=0$. The remaining eigenvalues and their corresponding eigenvectors characterize the spread of the beam in the hyperplane PL_t, i.e., its magnitude and direction.

5. 2. Mathematical model for detecting anomalous connections between computer system components

The dynamics of computer system behavior in the three-dimensional case are considered. This is a typical example of evaluating three main characteristics: processor load, memory load, network device load. To this end, we shall use the main provisions from the theory of dynamic chaos [13, 14].

When studying the spread of random trajectories around a cycle, one can use a visual geometric description. Fig. 1 shows the points of intersection (asterisks) of random trajectories with the intersecting plane PL_t, orthogonal to the cycle at the point $\xi(t)$. The covariance of the distribution of these points,

given by the matrix $W(t)$ is a function of the stochastic sensitivity of the cycle. It is the only solution to the system (1) to (4).

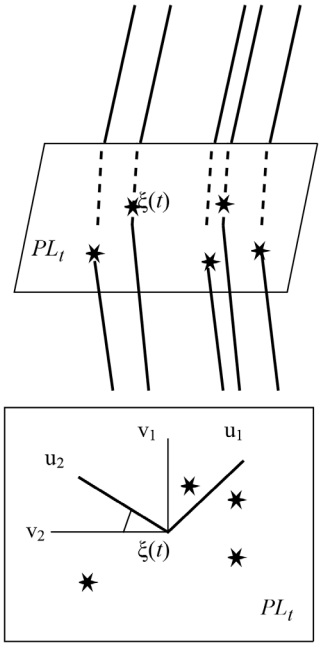

Fig. 1. Intersection points (asterisks) of random trajectories with the hyperplane PL_t orthogonal to the cycle at the point $\xi(t)$

In the studied three-dimensional case (n=3), we shall use the singular expansion to construct the solution $V(t)$ to equation (1):

$$V(t)=\lambda_1(t)v_1(t)v_1^T(t)+$$
$$+\lambda_2(t)v_2(t)v_2^T(t)+\lambda_3(t)v_3(t)v_3^T(t),$$

where $\lambda_1(t)\geq\lambda_2(t)\geq\lambda_3(t)$ are the eigenvalues, and $v_1(t)$, $v_2(t)$, $v_3(t)$ are the eigenvectors of the matrix $V(t)$. It follows from condition (3) that for any t the matrix $V(t)$ is degenerate (the distribution of intersection points is concentrated in the plane) PL_t. This means that $\lambda_3(t)=0$ and the corresponding eigenvector $v_3(t)=r(t)/\|r(t)\|$ is tangent to the cycle. As a result, the expansion of the matrix $V(t)$ takes the following form:

$$V(t)=\lambda_1(t)v_1(t)v_1^T(t)+\lambda_2(t)v_2(t)v_2^T(t). \quad (5)$$

Here $V(t)$ is given by scalar functions $\lambda_1(t)$, $\lambda_2(t)$ and vectors $v_1(t)$, $v_2(t)$. In the case of non-degenerate noises, the functions $\lambda_1(t)$, $\lambda_2(t)$ are strictly positive and determine at any t the variance in random trajectories of the cycle along vectors $v_1(t)$, $v_2(t)$. The $\lambda_1(t)$, $\lambda_2(t)$ values specify the size, and $v_1(t)$, $v_2(t)$ specify the direction of the axes of the scattering ellipse of the points of intersection of random trajectories with the plane PL_t. The equation of this ellipse in the plane PL_t takes the following form:

$$\left(x-\xi(t)\right)^T W^+(t)\left(x-\xi(t)\right)=2k^2,$$

where the parameter k sets the confidence probability $P=1-e^{-k}$.

We denote by $u_1(t)$, $u_2(t)$ some orthonormal basis of the plane PL_t. This basis can be easily found from the known τ-periodic solution $\xi(t)$. The eigenvectors $v_1(t)$, $v_2(t)$ can be obtained by rotating the basis $u_1(t)$, $u_2(t)$ at some angle $\varphi(t)$:

$$v_1(t)=u_1(t)\cos\varphi(t)+u_2(t)\sin\varphi(t), \quad (6)$$

$$v_2(t)=-u_1(t)\sin\varphi(t)+u_2(t)\cos\varphi(t). \quad (7)$$

As a result, the expansion of (5) to (7) makes it possible to express the unknown solution to the system (1) to (3) in terms of three scalar functions $\lambda_1(t)$, $\lambda_2(t)$, $\varphi(t)$.

It is denoted:

$$P_1(t)=v_1(t)v_1^T(t),\ P_2(t)=v_2(t)v_2^T(t).$$

It is noted that $P_i(t)$ (i=1, 2) are projection matrices:

$$P_iv_i=v_i,\ P_iv_i=0(i\neq j),\ P=P_1+P_2. \quad (8)$$

The expansion of (5) is represented in the form:

$$V(t)=\lambda_1(t)P_1(t)+\lambda_2(t)P_2(t).$$

The following assumption is accepted. The following identities hold for orthonormal vector functions $v_i(t)$ and projection matrices $P_i(t)=v_i(t)v_i^T(t)$ (i=1,2):

$$v_1^T(t)\widehat{P}_1(t)v_1(t)\equiv 0, \quad (9)$$

$$v_1^T(t)\widehat{P}_2(t)v_1(t)\equiv 0, \quad (10)$$

$$v_2^T(t)\widehat{P}_1(t)v_2(t)\equiv 0, \quad (11)$$

$$v_2^T(t)\widehat{P}_2(t)v_2(t)\equiv 0, \quad (12)$$

$$v_1^T(t)\widehat{P}_1(t)v_2(t)=\hat{\varphi}(t)+\hat{u}_1^T(t)u_2(t). \quad (13)$$

$$v_1^T(t)\widehat{P}_2(t)v_2(t)=\hat{\varphi}(t)+\hat{u}_1^T(t)u_2(t). \quad (14)$$

It is possible to prove this assumption as follows. Identity (9) follows directly from the relation:

$$v_1^T\widehat{P}_1v_2=v_1^T\left[v_1\hat{v}_1^T\right]v_1=v_1^T\left[v_1\hat{v}_1^T+\hat{v}_1v_1^T\right]v_1=$$
$$=v_1\hat{v}_1^T+\hat{v}_1v_1^T=\left[v_1v_1^T\right]\equiv 0. \quad (15)$$

Identity (12) is proved in the same way. Identity (10) follows from:

$$v_1^T\widehat{P}_2v_2=v_1^T\left[v_2\hat{v}_2^T+\hat{v}_2v_2^T\right]v_1=v_2v_1^Tv_1\hat{v}_2^T+\hat{v}_2v_1^Tv_1v_2^T\equiv 0. \quad (16)$$

Identity (11) is proved in the same way.

Using equations:

$$\hat{v}_1=\hat{u}_1\cos\varphi+\hat{u}_2\sin\varphi+v_2\hat{\varphi},$$

$$\hat{u}_1^Tu_1\equiv 0,\ \hat{u}_2^Tu_2\equiv 0,\ -\left(\hat{u}_1^Tu_2\right)=u_1^Tu_2.$$

After that, one can form the following expression:

$$v_1^T\widehat{P}_1v_2=v_1^T\left[v_1\hat{v}_1^T+\hat{v}_1v_1^T\right]v_2=v_2\hat{v}_1^T=$$
$$=\left(\hat{u}_1^T\cos\varphi+\hat{u}_2^T\sin\varphi+v_2^T\hat{\varphi}\right)v_2=\left(\hat{u}_1^T\cos\varphi+\hat{u}_2^T\sin\varphi\right)v_2+\hat{\varphi}=$$
$$=-\begin{pmatrix}\hat{u}_1^Tu_1\cos\varphi\sin\varphi+\hat{u}_1^Tu_2\cos^2\varphi-\\-\hat{u}_2^Tu_1\sin^2\varphi+\hat{u}_2^Tu_2\cos\varphi\sin\varphi+\hat{\varphi}=\hat{u}_1^Tu_2+\hat{\varphi}\end{pmatrix}.$$

(13) follows from these relations. Identity (14) is proved similarly.

The proven assumption provides a set of identities for orthonormal vector functions and projection matrices. These identities make it possible to significantly simplify calculations related to the analysis of sensitivity of cycles and interactions between vector projections in a complex technical system. Since the $P_i(t)$ matrices are projective and orthogonal, their use helps separate vector analysis in different directions, allowing for more efficient modeling and calculation.

Using projection matrices and orthogonal vectors to study the behavior of system components is a different approach from existing anomaly analysis models, such as correlation analysis or clustering-based analysis. Model (9) to (16) provides an opportunity to evaluate changes in the orthogonality of vectors, which makes it possible to detect complex interactions between components.

Orthogonal projections are the basis for detecting anomalous deviations that may indicate new or unexpected connections between processes. For example, lost orthogonality indicates a violation of component independence, which can be a sign of intrusion or malfunction.

Also, the model makes it possible to evaluate how different components of the system interact with each other. For example, a normal system state assumes that certain components (such as CPU usage and network activity) are weakly correlated. Using the model, it is possible to determine the presence of undesirable interactions between system components through the analysis of projection matrices and their behavior over time. The appearance of correlations between vectors that should be orthogonal can signal an unauthorized access attempt or the introduction of a new malicious process that affects several parts of the system at the same time.

Fig. 2 shows operational results of the model for detecting anomalies in a computer system. The plot displays the distribution of normal state (blue points) and abnormalities (red points) in 3D space.

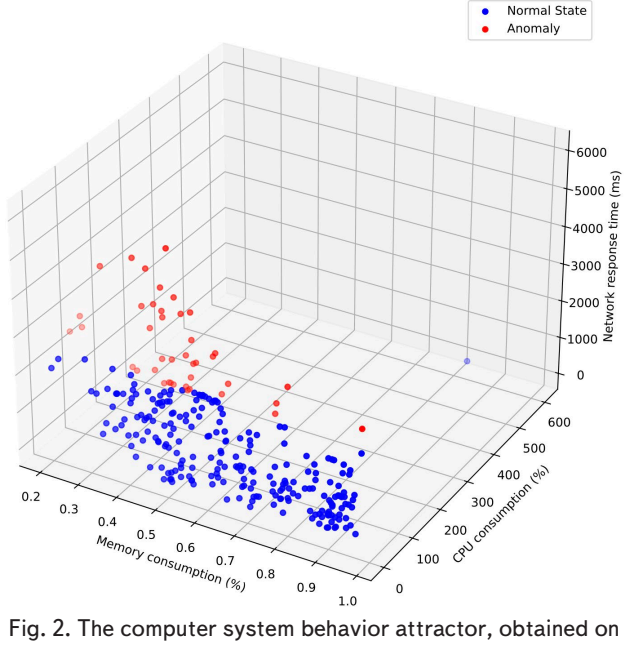

Fig. 2. The computer system behavior attractor, obtained on the basis of a model for detecting anomalous connections between computer system components

Table 1 gives values of the main indicators for the accuracy of detecting anomalies in a computer system.

Table 1

Indicators of accuracy of simulation results

No.	Accuracy indicator	Value	Execution time
1	Accuracy	0.84	0.00231 ms
2	Precision	0.87	
3	Recall	0.74	
4	F1 Score	0.78	

The plot in Fig. 2 and Table 1 demonstrate that this method selects some anomalous points, but they are largely intertwined with normal ones, which indicates the presence of errors in the identified anomalies. This may be a consequence of the unpreparedness of the data obtained from the CSE-CIC-IDS 2018 resource [15, 16]. In addition, it should be noted that using the built model in combination with machine learning models has the potential to improve accuracy indicators.

In order to check the possibility of improving the built model with the help of known machine learning methods, it should be noted that the main model for evaluating data for anomalies is the developed model for detecting anomalous connections between computer system components.

Fig. 3 shows 3D plots based on the results of the experiment using the built model, as well as machine learning models (Isolation Forest, Autoencoder, One-Class SVM) [17–20].

On the Isolation Forest plot, Fig. 3, one can see that the method detects anomalies distributed along all axes, with a lower concentration compared to normal points.

However, the accuracy of the method was not very high (~0.88). This indicates that some anomalies are not detected by the method, and part of the normal data may be mistakenly classified as anomalies.

The Autoencoder plot in Fig. 3 shows that the anomalies are fairly evenly distributed, but the method misses many anomalous points.

Performance metrics also did not show high values. This may indicate that the autoencoder has not fully learned the features of the system and may need further tuning of hyperparameters or a larger amount of training data, which is a difficult task under real-time conditions.

The plot of One-Class SVM in Fig. 3 demonstrates that the model performs better in some respects; however, the plot shows that there are areas where the model either misses anomalies or misclassifies normal data.

Table 2 gives values of the accuracy indicators of the considered examples.

As can be seen from the results in Tables 1, 2, the best time characteristics were shown by the built model for detecting anomalous relationships between computer system components and One-Class SVM. This indicates the possibility of using these models in real-time computer systems. At the same time, the built model has certain advantages in execution time, and the accuracy of anomaly detection is comparable to machine learning models.

Thus, a model for detecting anomalous connections between computer system components has been developed. The model is an advanced tool for mathematical analysis of the behavior of computer systems, especially in the context of detecting anomalous relationships between system components.

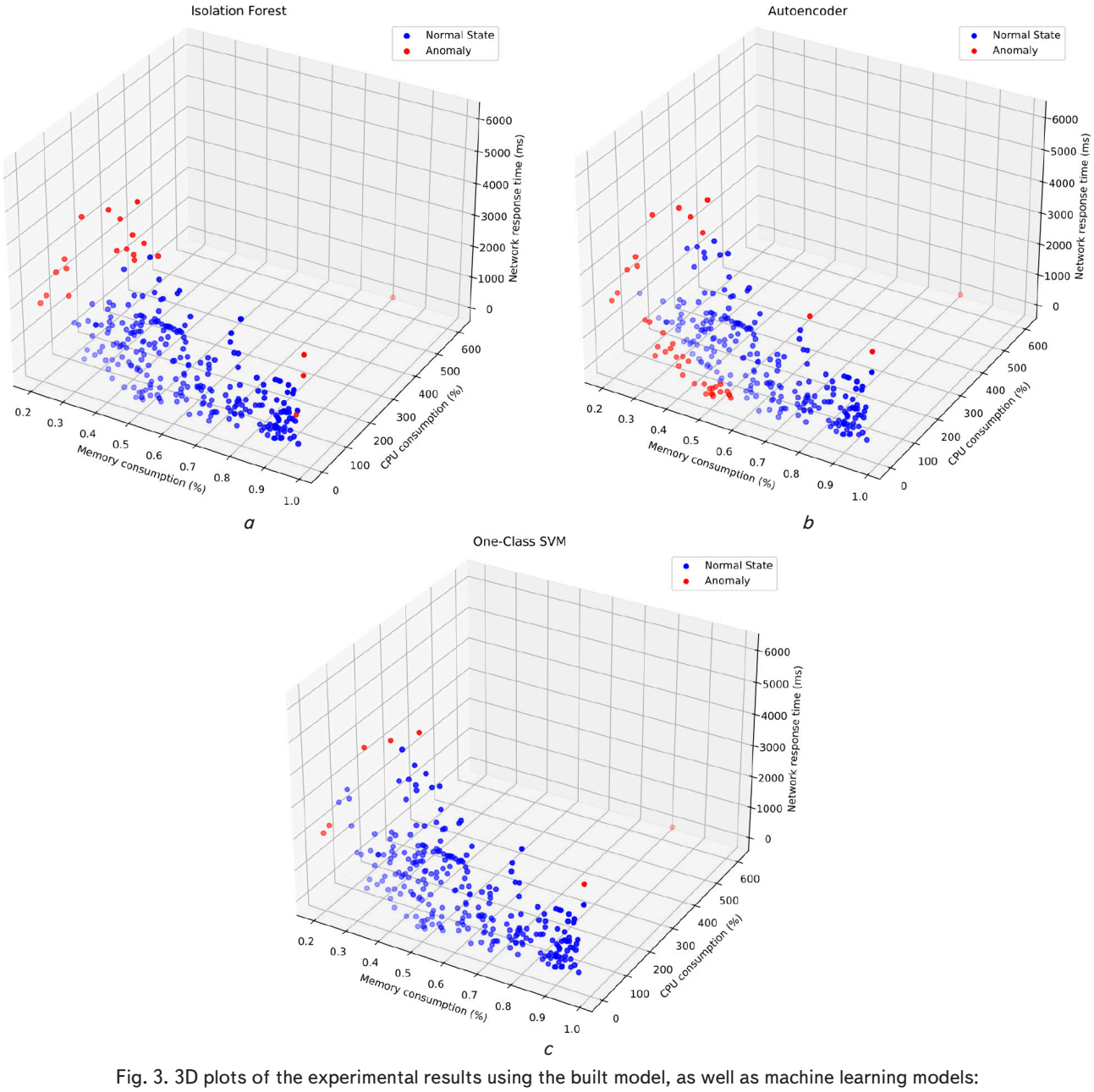

Fig. 3. 3D plots of the experimental results using the built model, as well as machine learning models: *a* – Isolation Forest, *b* – Autoencoder, *c* – One-Class SVM

Table 2

Accuracy indicators of simulation results using machine learning models

No.	Accuracy indicator	Value	Execution time
Isolation Forest			
1	Accuracy	0.95	787.361 ms
2	Precision	0.89	
3	Recall	0.65	
4	F1 Score	0.78	
Autoencoder			
1	Accuracy	0.93	33,020.259 ms
2	Precision	0.87	
3	Recall	0.73	
4	F1 Score	0.78	
One-Class SVM			
1	Accuracy	0.99	5.116 ms
2	Precision	0.99	
3	Recall	0.76	
4	F1 Score	0.88	

The model uses projection matrices and orthogonal vector functions to analyze anomalies. This makes it possible to create spatial layouts that make it possible to reveal complex relationships between components of a computer system, using only eigenvalues and vectors. This approach combines mathematical formalism with real technical data, which makes it an effective tool for deep analysis of system behavior.

The proposed model is distinguished by the use of eigenvalues and orthogonal vectors to construct ellipses of dispersion of trajectories of random processes in the system. This makes it possible to qualitatively evaluate and visualize the distribution of abnormal and normal states. In addition, the model makes it possible to describe behavior of the system as a geometric ellipse in space, in which parameters of the ellipse correspond to the level of loading of the processor, memory, and other resources, which provides the possibility of a detailed analysis of the relationships between the system components.

The next task of modeling is the mathematical formalization of the process of assessing the state of a computer system at a current time point or at short time intervals.

5. 3. Mathematical model for assessing the state of a computer system at a current time point or at short time intervals

It is assumed that the matrix $V(t)$ is a solution to the system (1) to (3) if and only if the scalar functions $\lambda_1(t)$, $\lambda_2(t)$, $\varphi(t)$, included in the expansion (5) to (7), satisfy the system:

$$\hat{\lambda}_1 = \lambda_1 v_1^T \left[F + F^T\right] v_1 + v_1^T S v_1, \tag{17}$$

$$\hat{\lambda}_2 = \lambda_2 v_2^T \left[F + F^T\right] v_2 + v_2^T S v_2, \tag{18}$$

$$\left(\lambda_1 - \lambda_2\right)\hat{\varphi} =$$

$$= \lambda_2 v_1^T F v_2 + \lambda_1 v_1^T F^T v_2 + v_1^T S v_2 - \left(\lambda_1 - \lambda_2\right) u_2 \hat{u}_1^T. \tag{19}$$

This statement is proved as follows. Let the matrix $V(t)$ be the solution to the system (1) to (3). Substituting the expansion $V=\lambda_1 P_1+\lambda_2 P_2$ into equation (1) leads to the following expression:

$$\hat{V} = \lambda_1 \hat{P}_1 + \hat{\lambda}_1 P_1 + \lambda_2 \hat{P}_2 + \hat{\lambda}_2 P_2 =$$

$$= \lambda_1 F P_1 + \lambda_2 F P_2 + \lambda_1 F^T P_1 + \lambda_2 F^T P_2 + \left(P_2 + P_1\right) S \left(P_2 + P_1\right).$$

Multiplying this ratio on the left by v_i^T and on the right by v_j, and using the properties of projection matrices (8) and the model (9) to (16), we obtain:

$$v_1^T V v_1 = \hat{\lambda}_1 = \lambda_1 v_1^T F v_1 + \lambda_1 v_1^T F^T v_1 + v_1^T S v_1, \tag{20}$$

$$v_2^T V v_2 = \hat{\lambda}_2 = \lambda_2 v_2^T F v_2 + \lambda_2 v_2^T F^T v_2 + v_2^T S v_2, \tag{21}$$

$$v_1^T V v_2 = \lambda_1 \left(\hat{\varphi} + \hat{u}_1^T u_2\right) + \lambda_2 \left(-\hat{\varphi} - \hat{u}_1^T u_2\right) =$$

$$= \lambda_2 v_1^T F v_2 + \lambda_1 v_1^T F^T v_2 + v_1^T S v_2. \tag{22}$$

Thus, $\lambda_1(t)$, $\lambda_2(t)$, $\mathrm{j}(t)$ satisfy system (17) to (19).

When mathematically modeling the process of assessing the state of a computer system at a current time point or at short time intervals, it is advisable to prove the opposite.

Let $\lambda_1(t)$, $\lambda_2(t)$, $\mathrm{j}(t)$ be the solutions to the system (6) to (19). Consider the matrix $V=\lambda_1 P_1+\lambda_2 P_2$. This matrix V satisfies condition (3). Due to (20) to (22), the following identities hold for $i, j=1, 2$:

$$v_i^T \left(\hat{V} - Q(V)\right) v_j \equiv 0. \tag{23}$$

Here $Q(V)=FV+VF^T+PSP$. After differentiation of the identity $r^T V_r \equiv 0$, we obtain:

$$\left[r^{\hat{T}} V r\right] = \hat{r}^T V r + r^T \hat{V} r + r^T V \hat{r} = r^T \hat{V} r \equiv 0,$$

$$r^T \left(\hat{V} - Q(V)\right) r \equiv 0.$$

Therefore, expression (23) also holds for $i, j=3$. It follows from expression (23) that the matrix V is a solution to equation (1). The assumption that was accepted at the beginning of the chapter turned out to be true.

As we can see, the construction of a solution to the system (1) to (3) based on the singular expansion (5), (6) is reduced to solving the system (20) to (22) for three scalar functions. The matrix $W(t)$ (the desired function of the stochastic sensitivity of the cycle) of the solution to the system (1) to (3) can be obtained from $V(t)$ using the following boundary transition:

$$\lim_{n\to\infty} \left(V(t) - W(t)\right) = 0. \tag{24}$$

One can see that the mathematical model (20) to (23) is focused on the estimation of the state of the system at the current time or on short time intervals. It describes the behavior of the sensitivity matrix, which changes under the influence of random factors in real time. Thus, this model can be used to detect current threats and anomalies occurring immediately or in the short term.

Based on this, the mathematical model (20) to (23) can be used to evaluate the computer system at the current time or at short time intervals. With the help of this model, deviations in the current trajectories of the system can be detected, which can be a sign of an attack attempt or external interference. This makes it possible to quickly react to threats, block suspicious processes, or perform isolation of potentially dangerous elements.

Also, this model makes it possible to estimate instantaneous changes in the sensitivity of the system to disturbances. For example, if the sensitivity matrix indicates an increased response to input data, this may be a signal of imminent danger or attack. This theorem helps quickly identify moments when a system becomes vulnerable and take measures for immediate protection.

Summarizing the preliminary results, it can be noted that the model is well suited for the development of rapid response systems, such as intrusion detection systems (IDS). It helps constantly monitor the state of the system and respond to threats in real time. This is especially important in situations where it is necessary to quickly respond to cyber attacks, for example, DDoS or hacking attempts.

Fig. 4 shows results of the computer system state assessment model at the current time or at short time intervals in the form of an attractor. As in the previous examples, the figure shows the distribution of the normal state (blue points) and anomalies (red points) in 3D space.

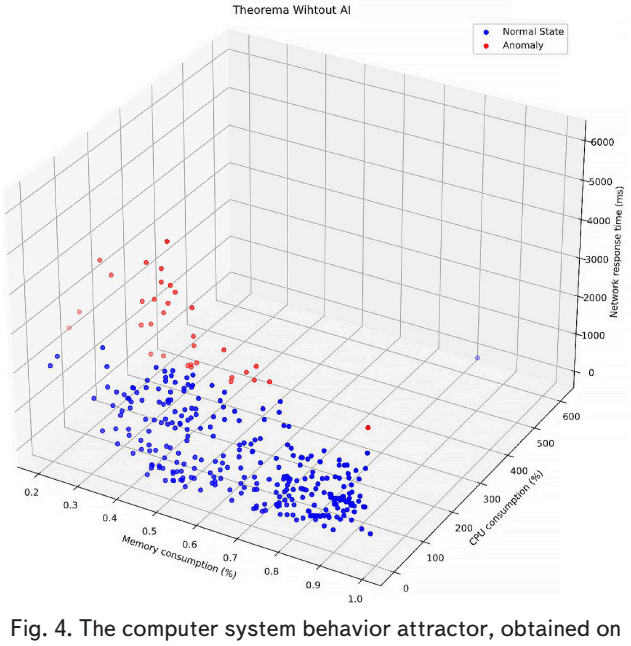

Fig. 4. The computer system behavior attractor, obtained on the basis of a model for assessing its state at the current time or at short time intervals

Table 3 gives values for the main indicators of accuracy in detecting anomalies in a computer system using the computer system state assessment model at the current time or at short time intervals.

Table 3

Accuracy indicators of simulation results using a computer system state assessment model at the current time or at short time intervals

No.	Accuracy indicator	Value	Execution time
1	Accuracy	0.91	0.198 ms
2	Precision	0.82	
3	Recall	0.68	
4	F1 Score	0.67	

Similar to the previous model, the plot in Fig. 4 and Table 3 demonstrate that this method selects some anomalous points. The results of the assessment using the "Accuracy" indicator show an increase in accuracy according to this indicator. At the same time, the results of the assessment of other indicators are practically identical to the results of the previous assessment model.

The possibility of improving the built model with the help of known methods of machine learning has been verified. In this case, the main model of data analysis for anomalies is the built model for assessing the state of a computer system at the current time point or at short time intervals.

Fig. 5 shows the 3D attractors of computer system behavior obtained on the basis of machine learning models (Isolation Forest, Autoencoder, One-Class SVM).

Table 4 gives values for the main indicators in the accuracy of detecting anomalies in a computer system using the integrated use of machine learning models and the model for assessing the state of a computer system at the current time or at short time intervals.

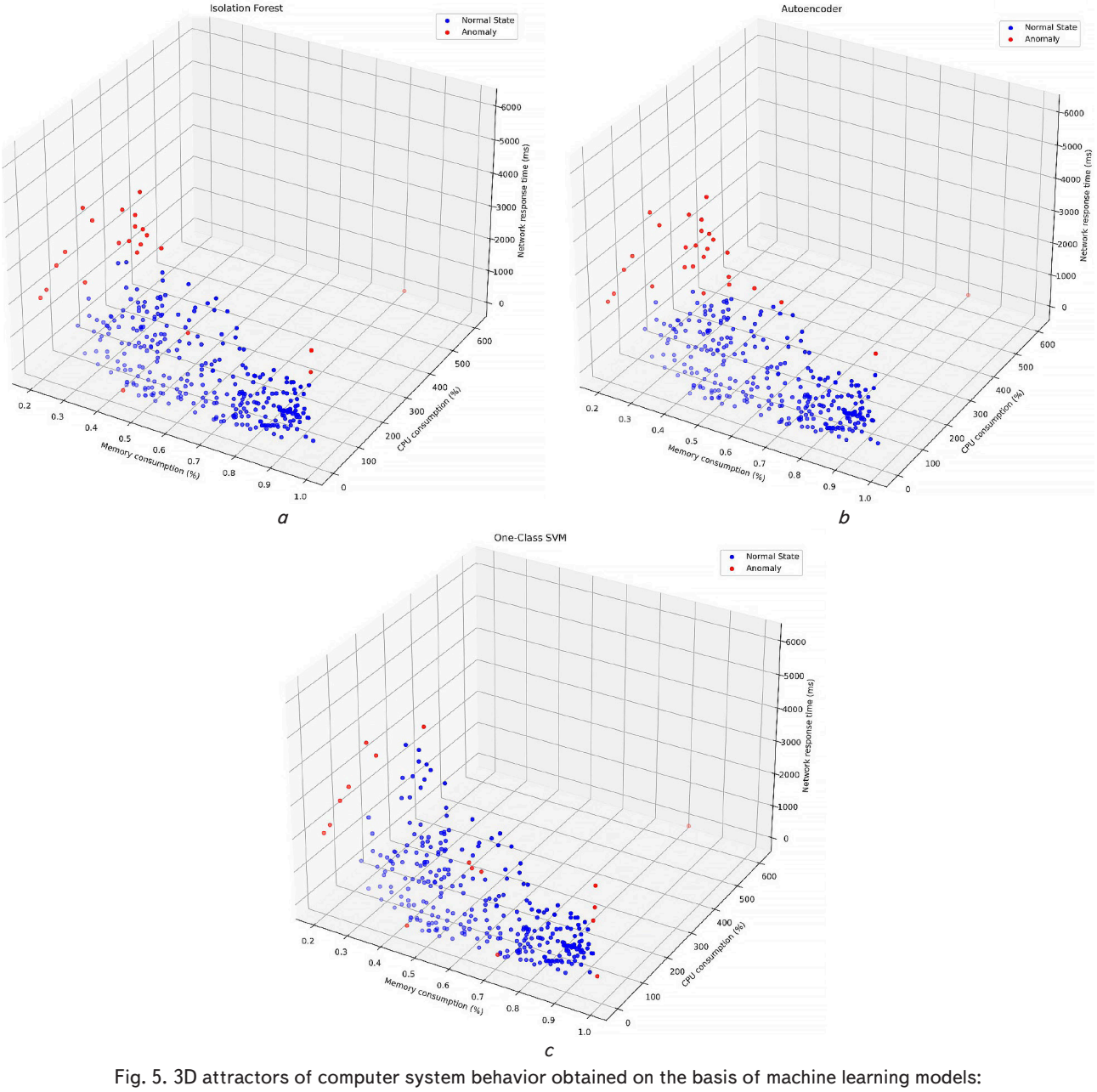

Fig. 5. 3D attractors of computer system behavior obtained on the basis of machine learning models: *a* – Isolation Forest; *b* – Autoencoder; *c* – One-Class SVM

Table 4

Accuracy indicators of simulation results using machine learning models

No.	Accuracy indicator	Value	Execution time
Isolation Forest			
1	Accuracy	0.93	3,050.91 ms
2	Precision	0.84	
3	Recall	0.64	
4	F1 Score	0.67	
Autoencoder			
1	Accuracy	0.91	97,711.45 ms
2	Precision	0.89	
3	Recall	0.73	
4	F1 Score	0.76	
One-Class SVM			
1	Accuracy	0.93	7.5361 ms
2	Precision	0.84	
3	Recall	0.63	
4	F1 Score	0.66	

Thus, the results of analyzing the execution time by each method for detecting anomalies can be summarized.

Isolation Forest uses decision trees to find anomalies, which explains its relatively long runtime. The algorithm is effective but requires quite significant computing resources. Its execution time is faster than the autoencoder, but longer compared to the One-Class SVM and the model-based approach.

Autoencoder has the highest execution time among all methods presented. This is explained by the complexity of the neural network, the number of parameters, and the need for long-term training of the model. Computational overhead makes autoencoders less suitable for systems where speed is important, but they can be useful in tasks with high accuracy requirements.

One-Class SVM showed a small execution time due to the efficiency of SVM for a limited amount of data. This method is suitable for some cases where speed of execution is critical.

Our mathematical model is the best option from the point of view of the execution time of the task of evaluating the anomalous behavior of the system. This makes the model suitable for real-time systems where minimum latency is important.

All three models have high Accuracy values, showing their capability to correctly classify most of the data.

Precision is higher in Autoencoder, indicating a better capability of this model to distinguish between true and false anomalies.

Recall is higher in Autoencoder (0.73), which means that this model detects more of all available anomalies compared to other models.

F1 Score, an indicator that balances between precision and recall, is also higher in Autoencoder. This indicates that this model generally performs better at the classification task compared to Isolation Forest and One-Class SVM.

All values presented are within the acceptable range for these anomaly detection methods. However, it should be noted that a high value of Accuracy with a relatively low Recall may indicate an imbalance in the data, where most examples belong to the normal class. This results in the model classifying most of the data as normal, which increases Accuracy but may miss a significant number of anomalies.

Precision and Recall vary between models, and this is normal because each model handles anomaly detection and response to outliers differently.

It should be noted that the accuracy indicators of the built model for assessing the state of a computer system at a current time point under the given conditions of the experiment showed a comparable result in comparison with the machine learning models. Of course, artificial intelligence models that have undergone a certain learning path would show better accuracy results. But the time characteristics of these models would still be worse compared to the model built.

6. Discussion of results of investigating mathematical models

Our results indicate that the proposed mathematical models are capable of detecting anomalies in highly loaded complex computer systems in real time. This can be explained by the features of the built models, which use sensitivity matrices to analyze the dynamics of the system – expressions (5) to (16) and (20) to (23). The theoretical substantiation of the model makes it possible to reduce the number of false positives and ensure stable operation of the system under conditions of high load. In addition, the use of projection methods significantly increased the ability of the model to quickly respond to changes in input parameters. This is confirmed by the results of the assessment of quality indicators in Fig. 2 and Table 2.

In contrast to models (4) to (8), the advantage of the proposed approach is the ability to evaluate changes in the orthogonality of vectors, which makes it possible to detect complex interactions between components and provide the ability to detect anomalies in a short time. The mathematical apparatus used in the model makes it possible to reduce the number of necessary calculations, which is important for systems with limited resources. Unlike many existing methods, such as autoencoders or machine learning methods [9, 17, 18], which require significant computing resources and a large amount of training data, our approach is based on a rigorous theoretical foundation and has the potential to work effectively in real time.

The resulting solutions are aimed at resolving the issue related to increasing the speed of anomaly detection, taking into account the accuracy requirements for HLCCS. One of the main problems was the impossibility of effectively detecting anomalies in real time under conditions of large volumes of data and complex system dynamics. The results show that the built model is able to solve this problem thanks to the use of dynamic chaos theory and sensitivity matrices. Compared to models [4–7], this approach allows models to accurately and quickly respond to changes in the system.

Our research into the integrated use of built models with machine learning models described in the literature [9–19] allowed us to draw a conclusion about the sensitivity of this approach to the quality of input data. If the data contains a significant amount of noise or errors, it can negatively affect the results of the anomaly detection. In addition, the model may require additional adaptation for use under specific conditions of particular computer systems, which may require additional costs for calibration. And this research result confirms the need for further improvement of the complex modeling approach.

In addition, another disadvantage of the proposed approach is its limited ability to work with heterogeneous data,

which are characterized by significant heterogeneity and dynamics. In future studies, it is planned to expand the model by adding the possibility of working with multidimensional and heterogeneous data. There is also a need for additional testing on different platforms to assess the portability of the proposed solutions.

The limitations of our study are that the results of the simulation are largely dependent on the quality of the input data. If the data contains a significant amount of noise or errors, it can negatively affect the results of the anomaly detection. In the study, data from a proven source [16] CSE-CIC-IDS2018 on AWS was selected to improve the quality of anomaly detection results.

The model built can be used to ensure the reliability and security of highly loaded computer systems, such as banking transaction servers or telecommunication platforms. Real-time anomaly detection prevents possible failures and attacks, which is critical for the continuous operation of such systems.

7. Conclusions

1. As a result of our research, a set of mathematical models was built for detecting anomalies in highly loaded complex computer systems. The main feature is the use of the provisions from the theory of dynamic chaos, which made it possible to increase the accuracy and speed of anomaly detection. The models successfully identify small anomalies that may indicate hidden problems or instability of the system, as well as critical deviations that pose a threat to its security. This provides an opportunity to promptly respond to potential threats and increase the stability of the system.

2. A mathematical model for detecting anomalous connections between computer system components has been developed, which, unlike others, uses a geometric approach, where anomalies are detected through a change in mutual orthogonality between components. That made it possible to reduce the time of detecting anomalies in the state of a computer system by up to 10 %. At the same time, the accuracy of anomaly detection remained at the predefined level.

3. A mathematical model for assessing the state of a computer system at a current time point or at short time intervals has been built. Unlike others, it has the capability to reflect temporal changes in relationships between various components of the system. This is especially important for highly loaded, complex computer systems where changes can have a large impact on stability and security. The mathematical model built makes it possible not only to detect existing anomalies but also evaluate future scenarios based on dynamic behavior, which provides a proactive approach to risk management and system security. A study of the use of the built models in combination with the Isolation Forest, Autoencoder, One-Class SVM models was conducted. The results of the research showed a significant (up to 10 times) increase in the speed of detecting anomalies in the behavior of the computer system, with a slight decrease in the accuracy of this operation. This makes it possible to draw conclusions about the expediency of using the built models to detect anomalies in the behavior of highly loaded complex computer systems in real time.

Conflicts of interest

The authors declare that they have no conflicts of interest in relation to the current study, including financial, personal, authorship, or any other, that could affect the study, as well as the results reported in this paper.

Funding

The study was conducted without financial support.

Data availability

All data are available, either in numerical or graphical form, in the main text of the manuscript.

Use of artificial intelligence

The authors confirm that they did not use artificial intelligence technologies when creating the current work.

References

1. Yu, S., Jiang, H., Huang, S., Peng, X., Lu, A. (2021). Compute-in-Memory Chips for Deep Learning: Recent Trends and Prospects. IEEE Circuits and Systems Magazine, 21 (3), 31–56. https://doi.org/10.1109/mcas.2021.3092533
2. Kumar, S., Gupta, S., Arora, S. (2021). Research Trends in Network-Based Intrusion Detection Systems: A Review. IEEE Access, 9, 157761–157779. https://doi.org/10.1109/access.2021.3129775
3. Lu, P.-J., Lai, M.-C., Chang, J.-S. (2022). A Survey of High-Performance Interconnection Networks in High-Performance Computer Systems. Electronics, 11 (9), 1369. https://doi.org/10.3390/electronics11091369
4. Semenov, S., Mozhaiev, O., Kuchuk, N., Mozhaiev, M., Tiulieniev, S., Gnusov, Y. et al. (2022). Devising a procedure for defining the general criteria of abnormal behavior of a computer system based on the improved criterion of uniformity of input data samples. Eastern-European Journal of Enterprise Technologies, 6 (4 (120)), 40–49. https://doi.org/10.15587/1729-4061.2022.269128
5. Meleshko, Y., Raskin, L., Semenov, S., Sira, O. (2019). Methodology of probabilistic analysis of state dynamics of multidimensional semiMarkov dynamic systems. Eastern-European Journal of Enterprise Technologies, 6 (4 (102)), 6–13. https://doi.org/10.15587/1729-4061.2019.184637
6. Semenov, S., Zhang, L., Cao, W., Bulba, S., Babenko, V., Davydov, V. (2021). Development of a fuzzy GERT-model for investigating common software vulnerabilities. Eastern-European Journal of Enterprise Technologies, 6 (2 (114)), 6–18. https://doi.org/10.15587/1729-4061.2021.243715

7. Meleshko, Y., Yakymenko, M., Semenov, S. (2021). A Method of Detecting Bot Networks Based on Graph Clustering in the Recommendation System of Social Network. International Conference on Computational Linguistics and Intelligent Systems. Available at: https://ceur-ws.org/Vol-2870/paper92.pdf
8. Semenov, S., Gavrylenko, S., Chelak, V. (2016). Developing parametrical criterion for registering abnormal behavior in computer and telecommunication systems on the basis of economic tests. Actual problems of economics, 4 (178), 451–459.
9. Angel, N. A., Ravindran, D., Vincent, P. M. D. R., Srinivasan, K., Hu, Y.-C. (2021). Recent Advances in Evolving Computing Paradigms: Cloud, Edge, and Fog Technologies. Sensors, 22 (1), 196. https://doi.org/10.3390/s22010196
10. Khan, A. R. (2024). Dynamic Load Balancing in Cloud Computing: Optimized RL-Based Clustering with Multi-Objective Optimized Task Scheduling. Processes, 12 (3), 519. https://doi.org/10.3390/pr12030519
11. Zhao, L., Gao, W., Fang, J. (2024). Optimizing Large Language Models on Multi-Core CPUs: A Case Study of the BERT Model. Applied Sciences, 14 (6), 2364. https://doi.org/10.3390/app14062364
12. Dakić, V., Kovač, M., Slovinac, J. (2024). Evolving High-Performance Computing Data Centers with Kubernetes, Performance Analysis, and Dynamic Workload Placement Based on Machine Learning Scheduling. Electronics, 13 (13), 2651. https://doi.org/10.3390/electronics13132651
13. Savi, M. A. (2023). Chaos Theory. Lectures on Nonlinear Dynamics, 283–299. https://doi.org/10.1007/978-3-031-45101-0_10
14. Devaney, R. L. (2021). An Introduction to Chaotic Dynamical Systems. Chapman and Hall/CRC. https://doi.org/10.1201/9780429280801
15. Göcs, L., Johanyák, Z. C. (2024). Identifying relevant features of CSE-CIC-IDS2018 dataset for the development of an intrusion detection system. Intelligent Data Analysis, 28 (6), 1527–1553. https://doi.org/10.3233/ida-230264
16. CSE-CIC-IDS2018 on AWS. Available at: https://www.unb.ca/cic/datasets/ids-2018.html
17. Almansoori, M., Telek, M. (2023). Anomaly Detection using combination of Autoencoder and Isolation Forest. 1st Workshop on Intelligent Infocommunication Networks, Systems and Services, 25–30. https://doi.org/10.3311/wins2023-005
18. Ribeiro, D., Matos, L. M., Moreira, G., Pilastri, A., Cortez, P. (2022). Isolation Forests and Deep Autoencoders for Industrial Screw Tightening Anomaly Detection. Computers, 11 (4), 54. https://doi.org/10.3390/computers11040054
19. Gavrylenko, S. Y., Sheverdin, I. V. (2021). Development of method to identify the computer system state based on the "isolation forest" algorithm. Radio Electronics, Computer Science, Control, 1 (1), 105–116. https://doi.org/10.15588/1607-3274-2021-1-11
20. Semenov, S., Sira, O., Gavrylenko, S., Kuchuk, N. (2019). Identification of the state of an object under conditions of fuzzy input data. Eastern-European Journal of Enterprise Technologies, 1 (4 (97)), 22–30. https://doi.org/10.15587/1729-4061.2019.157085