Methods for Predicting Failures in a Smart Home

Viktoriia Zhebka[1], Pavlo Skladannyi[2], Yurii Bazak[1], Andrii Bondarchuk[1], and Kamila Storchak[1]

[1] State University of Information and Communication Technologies, 7 Solomenskaya str., Kyiv, 03110, Ukraine
[2] Borys Grinchenko Kyiv University, 18/2 Bulvarno-Kudriavska str., Kyiv, 04053, Ukraine

Abstract

Methods for predicting possible failures in smart home systems and analyzing the data required for this have been considered in the study. A study of machine learning methods has been carried out: their features, advantages, and disadvantages have been identified, the metrics of each method have been studied, and the effectiveness of methods for predicting failures in a smart home has been established. It has been found that the Long Short-Term Memory (LSTM) model is distinguished by its ability to work with data sequences and store information for a long time. The characteristics of the LSTM method and its algorithm have been studied in detail. The study emphasizes the importance of collecting and processing various data, such as sensor data, energy consumption, and information about devices and users. The results of the study can be useful for the further development of smart home control systems to improve their reliability and efficiency.

Keywords

Long short-term memory, LSTM, Machine learning, data processing, forecasting, smart home, failure, information technology.

1. Introduction

Smart houses are becoming increasingly common thanks to the development of the Internet of Things (IoT) and smart technologies [1]. They provide automation and ease of control of various systems such as lighting, heating, security, energy efficiency, and many others [2, 3].

However, as the complexity of these systems grows, the likelihood of failures or problems increases. Network instability, software errors, and faulty devices can all lead to unpredictable situations that affect the usability and security of a smart home [4].

Therefore, predicting failures in a smart home is a relevant issue, and it is appropriate to conduct such a prediction using machine learning methods. The use of machine learning algorithms allows for analyzing large amounts of data; and identifying deviations and patterns that precede failures. This approach allows predicting possible problems and taking measures to prevent them before they occur.

Today, smart home failure prediction is mostly based on reactive data analysis. This means that systems detect anomalous situations or failures after they occur, which can make it difficult to avoid potential problems.

However, using machine learning methods, such as classification, clustering, and prediction algorithms, it is possible to develop systems that can predict failures in a smart home in advance.

Such systems use data analysis from sensors, IoT devices, control systems, energy consumption, and other data to identify patterns and anomalies that may precede disruptions [5, 6]. Based on this information, machine learning systems can build predictive models that respond to certain signals or changes in normal operation, warning of potential problems or taking steps to prevent them [7, 8].

This area of research is still evolving, but it promises to improve smart home control

DECaT'2024: Digital Economy Concepts and Technologies, April 4, 2024, Kyiv, Ukraine
EMAIL: viktoria_zhebka@ukr.net (V. Zhebka); p.skladannyi@kubg.edu.ua (P. Skladannyi); jura.bazak@gmail.com (Y. Bazak); dekan.it@ukr.net (A. Bondarchuk); kpstorchak@ukr.net (K. Storchak)
ORCID: 0000-0003-4051-1190 (V. Zhebka); 0000-0002-7775-6039 (P. Skladannyi); 0009-0000-6098-2809 (Y. Bazak); 0000-0001-5124-5102 (A. Bondarchuk); 0000-0001-9295-4685 (K. Storchak)

systems by enabling them to predict and prevent possible failures in advance, providing greater reliability and security for users.

2. Research Results

Machine learning techniques can help detect and avoid some disruptions before they occur or restore system operations faster after a failure. They can provide early detection of anomalies in performance, which will prevent problems from occurring or respond quickly to them, which in turn will help reduce the impact of these failures on the smart home.

Machine learning algorithms are compared on four different data representations: original, balanced, normalized, and standardized. The original data is unchanged from the selected data except for the removal of timestamp values. Balancing is performed by under-sampling the faultless data in the training data. Failure-free data inputs are randomly selected and removed from the dataset, resulting in the same number of failures and no failures. Insufficient sampling can erase important information from the data, leading to poor algorithm performance. The main advantage of data balancing is that it reduces the resources and time required to train algorithms. Data normalization refers to the scaling of feature values in the range from zero to one. Scaling is performed using (1) separately for each feature by finding its minimum and maximum values. The data is also standardized by considering each feature separately. The feature values are subtracted from the mean and then divided by the standard deviation, as shown in (2). These results in values centered around zero with unit dispersion. An additional pre-processing step that is performed before normalizing and standardizing the data is the conversion of the time features of the day of the week and the hour. To indicate that the difference between the hours 23 and 0 is the same as the difference between 22 and 23, the values are converted to cyclic representations using the Fourier transform [6]. The transformation calculates the sine and cosine values for each feature, as shown in equation (3). Thus, each feature is replaced by the corresponding sine and cosine feature. The calculation depends on the total number of different feature values N, which is 24 for hours and 7 for days of the week.

$$x_n = \frac{x - \min}{\max - \min}. \tag{1}$$

$$x_n = \frac{x - \mu}{\sigma} \tag{2}$$

$$x_{\sin} = \sin\left(\frac{2\pi x}{N}\right), \qquad x_{\cos} = \cos\left(\frac{2\pi x}{N}\right) \tag{3}$$

where x is the feature value, min is the minimum value of the feature, max is the maximum value of the feature, μ is the average value of the feature, σ is the standard deviation of a feature, and N is the total number of different values of the features.

Some of the algorithms may have problems with dimensionality and run much slower than other algorithms. The problem can be solved by reducing the number of dimensions using principal component analysis. The method is only applicable to specific algorithms and specific data representations, depending on the speed of learning and prediction. In addition, some of the algorithms work only with a certain input format.

Many machine learning algorithms can be used to predict device failures. They can differ in many properties and features [9]. They can be supervised or unsupervised, they can solve classification, regression, or clustering problems or they can belong to different families such as deep learning, tree-based, probabilistic, or linear. The total number of algorithms considered for comparison was limited due to their lengthy setup and training. The algorithms were selected based on several criteria:

1. Supervised learning: based on the assumption that the data to be used have been chosen.
2. Practical use: some of the algorithms are used more than others for predictive maintenance.
3. Diversity: algorithms were chosen to represent different families, tasks, and functions, such as online learning or prediction over time.

Based on these criteria, ten algorithms have been selected, nine of which are implemented as classification algorithms and one as a time series regression algorithm (Tables 1 and 2) [10, 11]. There are representatives of different types. All algorithms support online learning either implicitly or through certain variations.

Table 1
List of machine learning methods and their brief description

English name	Description
k-Nearest Neighbor	A classification method that determines the class of an object by analyzing its nearest neighbors in the feature space [12].
Decision Tree	A classification or regression algorithm that uses a tree structure to make decisions based on feature separations [13].
Random Forest	Decision tree ensemble, where multiple trees are combined to avoid overlearning and improve accuracy [14].
Extreme Gradient Boosting	An ensemble of models that use gradient lift to improve the accuracy of each subsequent model [15].
Naive Bayes	A probabilistic method that uses Bayes' theorem for classification based on the probabilities of features entering a class [16].
Support Vector Machine	An algorithm that determines the optimal boundary of separation between classes using support vectors [17].
Logistic Regression	A classification method that uses a logistic function to determine the probability of an object belonging to a certain class [12].
Stochastic Gradient Descent	An optimization algorithm that uses a gradient to find the minimum of a loss function with a randomly selected subset of data [16].
Multi-Layer Perceptron	A neural network with one or more hidden layers is used for classification and regression based on weighting coefficients [12].
LSTM	A type of recurrent neural network designed to store and use information over a long period to predict failures or events [16, 17].

Table 2
Comparative characteristics of machine learning methods

Method	The principle of operation	Application area	Advantages	Disadvantages
k-Nearest Neighbor	Determining the class of an object through its nearest neighbors	Detecting anomalies, predicting failures	Easy to implement, no training required	Sensitivity to emissions, high computational costs
Decision Tree	Decision-making based on sequential divisions by features	Anomaly detection, failure classification	Ease of interpretation, accommodating conditions	Tendency to overlearn, instability
Random Forest	Tree ensemble to avoid overtraining	Detecting anomalies, predicting failures	High accuracy, and consistency of solutions	A large number of hyperparameters, training time
Extreme Gradient Boosting	Using gradient lift to improve accuracy	Failure prediction, anomaly detection	High accuracy, less tendency to overlearn	A large number of hyperparameters, complexity of interpretation
Naive Bayes	Using Bayes' theorem for probabilistic classification	Filtering anomalies, detecting failure patterns	Efficiency for small data, simplicity of the model	The predictions are not flexible enough
Support Vector Machine	Determining the optimal boundary of separation between classes	Classification of anomalies, forecasting failures	Efficiency in large spaces, flexibility	Requirements for data preparation, high complexity of customization
Logistic Regression	Determining the probability of an object belonging to a certain class	Failure classification, anomaly detection	Interpretability, ease of implementation	Requires a linear separation surface
Stochastic Gradient Descent	Using a gradient to optimize the loss function	Model training, failure analysis	Fast learning, efficient for big data	Requirements for hyperparameters, tendency to stutter
Multi-Layer Perceptron	A neural network with one or more hidden layers	Pattern recognition, time series forecasting	Ability to solve complex problems	Requires a lot of data for training, training time
LSTM	Recurrent neural network for long-term memorization	Time sequence analysis, failure prediction	Ability to recognize dependencies over time	High computational costs, the complexity of setup

The main metrics for evaluating different machine learning algorithms in prediction or classification tasks allow for an objective comparison of the effectiveness of these algorithms.

Accuracy represents the percentage of correctly classified cases in the total number of cases, which gives a general idea of the algorithm's accuracy [11]:

$$T = (TP + TN) / (FP + FN + TP + TN), \quad (4)$$

where *TP* is true positive (correctly categorized positive ones), *TN* is true negative (correctly categorized negative ones), *FP* is false positive (incorrectly categorized positive ones), and *FN* is false negative (incorrectly categorized negative ones).

The precision determines the percentage of correctly identified positive classes among all identified positive classes, which is useful for working with uneven classes.

$$P = \frac{TP}{FP + TP}. \quad (5)$$

Recall displays the percentage of correctly identified positive classes among all actual positive classes, which is important for identifying important cases that have been missed.

$$R = \frac{TP}{(TP + FN)}. \quad (6)$$

F1-average is a score that uses the harmonic mean between accuracy and completeness to understand how well the model solves the classification task.

$$F1 = 2\frac{PR}{P + R}. \quad (7)$$

ROC-AUC measures the area under the ROC curve and evaluates the model's performance depending on different classification thresholds, helping to determine its ability to make correct predictions.

$$ROC - AUC = \int_0^1 R(S)\,\mathrm{d(S)}. \quad (8)$$

To approximate this area, numerical methods such as the trapezoidal method or Simpson's method are usually used, since the ROC-AUC formula itself is an integral.

The completeness (*R*) and frequency (*S*) are determined using a confusion matrix for binary classification.

Indicator *S* is calculated using formula (6), and the frequency is calculated using formula (9).

$$S = \frac{FP}{FP + TN}. \quad (9)$$

The Confusion Matrix provides detailed information about the real and predicted classes, which helps to estimate the level of correctness and errors for each class, which is important when analyzing the model.

The Confusion Matrix helps to evaluate the performance of a classification model by visualizing real and predicted values. It is the basis for calculating various metrics, such as accuracy, sensitivity, specificity, F1 score, and others.

These metrics are crucial for evaluating the effectiveness of algorithms and choosing the one that is most suitable for a particular task, depending on the requirements and needs [18, 19].

Table 3 shows the performance of different machine learning methods by the main metrics considered.

Table 3 and Fig. 1 show the performance of different machine learning methods in terms of the main metrics such as precision, accuracy, classification accuracy, completeness, F1-mean, and ROC-AUC. The score of "High," "Medium", or "Very High" in the "Performance" column is a generalized characterization of the methods' performance based on these metrics.

Table 3

Effectiveness of different machine learning methods by key metrics

Method	Accuracy	Classification accuracy	Completeness	F1-average	ROC-AUC	Effectiveness
k-Nearest Neighbor	0.85	0.81	0.89	0.85	0.92	High
Decision Tree	0.78	0.82	0.75	0.76	0.85	High
Random Forest	0.81	0.85	0.79	0.80	0.88	High
Extreme Gradient Boosting	0.87	0.88	0.86	0.87	0.94	High
Naive Bayes	0.75	0.79	0.72	0.73	0.82	Average
Support Vector Machine	0.82	0.84	0.80	0.81	0.89	High
Logistic Regression	0.79	0.83	0.77	0.78	0.86	Average
Stochastic Gradient Descent	0.80	0.82	0.79	0.80	0.87	Average
Multi-Layer Perceptron	0.84	0.86	0.82	0.83	0.91	High
LSTM	0.88	0.90	0.87	0.88	0.95	Very high

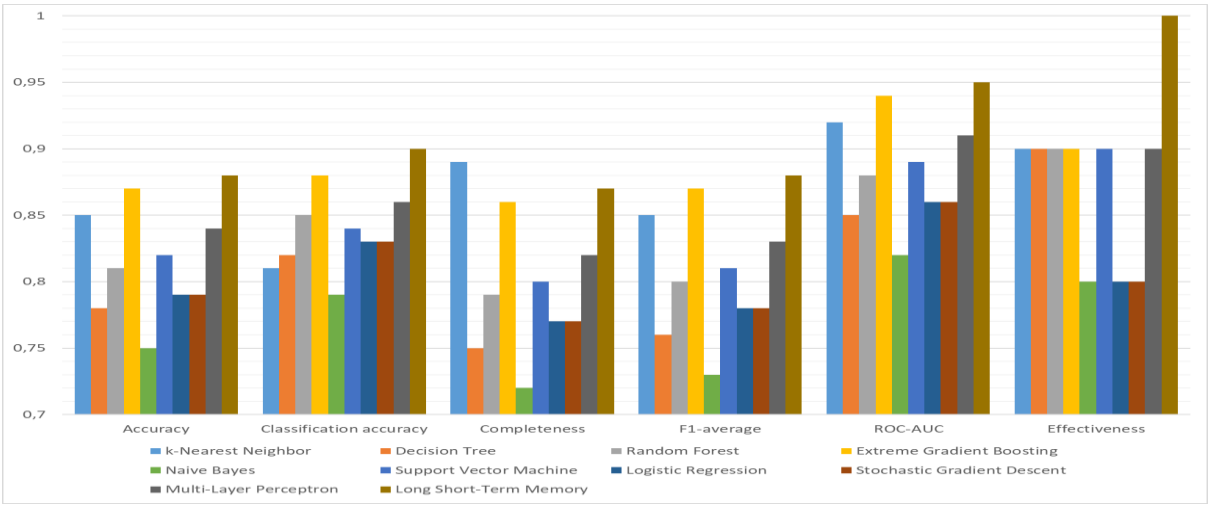

Figure 1: Comparison of machine learning methods

The study has found that the LSTM model is distinguished by its ability to work with data sequences and store information for a long time. This makes LSTM effective for analyzing time series, such as sensor data in a smart home, where information is usually sequential in time.

The LSTM model is capable of storing information for a long time, allowing it to effectively understand and analyze a sequence of real-time sensor data. By using mechanisms that ensure that some information is forgotten and others are retained, the LSTM can take into account long-term dependencies and the importance of individual events in time series. LSTM can adapt to and learn from different amounts of data, including large amounts of data from smart home sensors, which allows for more accurate predictions of failures. The LSTM model can adapt to changing conditions and detect changes in time series, which allows for predicting failures and anomalies in real-time. LSTM can process a variety of data types (text, numbers, sequences, etc.), making it versatile for use in various forecasting and analysis scenarios.

That is why it is not surprising that this algorithm showed the best results for predicting failures in a smart home.

The LSTM model has the following elements at each time step t:

1. Input data x_t
2. Previous output h_{t-1}
3. Previous memory state C_{t-1}
4. Gates:

- Forget gate f_t
- Input gate i_t
- Output gate o_t.

5. New memory state C_t
6. New exit h_t.

The model itself consists of the following formulas:

1. Forgetting the previous memory state:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right). \quad (10)$$

2. Defining what will be updated in memory:

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right). \quad (11)$$

$$\tilde{C}_t = \mathrm{tg}\left(W_C \cdot [h_{t-1}, x_t] + b_C\right). \quad (12)$$

3. Update the memory status:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t. \quad (13)$$

4. Update the output value:

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right). \quad (14)$$

$$h_t = o_t * \mathrm{tg}(C_t). \quad (15)$$

where f_t is the forget gate, which decides that the previous state should be forgotten, i_t is the input gate, which determines how much of the new input will be added to the memory state, $\tilde{C}_t$ is a new candidate for the memory state, C_t is memory status, o_t is output gate, determines which output will be next, x_t is input on a time step t, h_{t-1} is preliminary output on the time step $t-1$, W and a b is weights and displacements to be taught during the training process.

These equations allow the LSTM to determine what information to forget, what to keep, and how to use it to generate output values.

The step-by-step algorithm for training an LSTM model includes the following steps:

1. Data preparation:
- Input: receive a set of data containing time series or sequences.
- Data preparation: normalization, and conversion of data format to meet model requirements.

2. Building an LSTM architecture:
- Creating a model: using machine learning libraries to build an LSTM network.
- Defining parameters: number of layers, number of neurons in each layer, activation functions, etc.

3. Data separation:
- Training and testing sets: splitting data into training and testing sets to evaluate model performance.

4. Model training:
- Model training: fitting LSTM to training data using backpropagation.
- Model evaluation: assessing whether the model's predictions match the actual data on the test set.

5. Evaluation and improvement of the model:
- Analyzing the results: reviewing the forecast results and assessing their accuracy.
- Improving the model: use optimization techniques, change hyperparameters, or modify the architecture to improve results.

6. Testing and forecasting:
- Failure prediction: applying a trained model to new data to predict failures in a smart home.

7. Evaluation of the results:
- Performance evaluation: comparing forecasts with real data to determine the accuracy and efficiency of the model.

8. Maintaining and improving the model:
- Continuous learning: collecting new data and improving the model based on it to keep forecasts up-to-date.

This is a general description of the LSTM learning algorithm for predicting failures in a smart home.

The approach presented in this study takes advantage of LSTM to predict time series. The LSTM is implemented using Keras (a high-level neural network interface that simplifies the process of creating and training artificial neural networks; it is a machine learning library that runs on the Tensorflow, Theano, and Microsoft Cognitive Toolkit frameworks) as a sequential model with two LSTM layers and a dense output layer. It receives a sequence of data inputs (normalized feature values without rejections) and outputs a sequence of failure values.

It was decided to use a single data input, as this significantly reduces the training time and is sufficient for the algorithm to recognize failure patterns. The length of the source sequence determines the duration of the runtime. Prediction performance is tested on three different input sequences: 1 (1 second), 300 (5 minutes), and 1800 (30 minutes). As a result, three different LSTM models were built. For training, we used a dataset with 70% failures, creating a split into training and test sets in the proportion of 25–75% without mixing. In addition, the data was prepared by creating output sequences for each data record, which were then used for training.

Based on the conducted research, a data prediction platform has been developed. Once the system has been successfully integrated into the smart home, the implementation process takes place, when the system becomes an active part of the home environment. However, this is only the beginning: further support, optimization, and continuous improvement of the system play a key role in ensuring its long-term and efficient operation in a smart home, adapting to changing needs and conditions (Fig. 2).

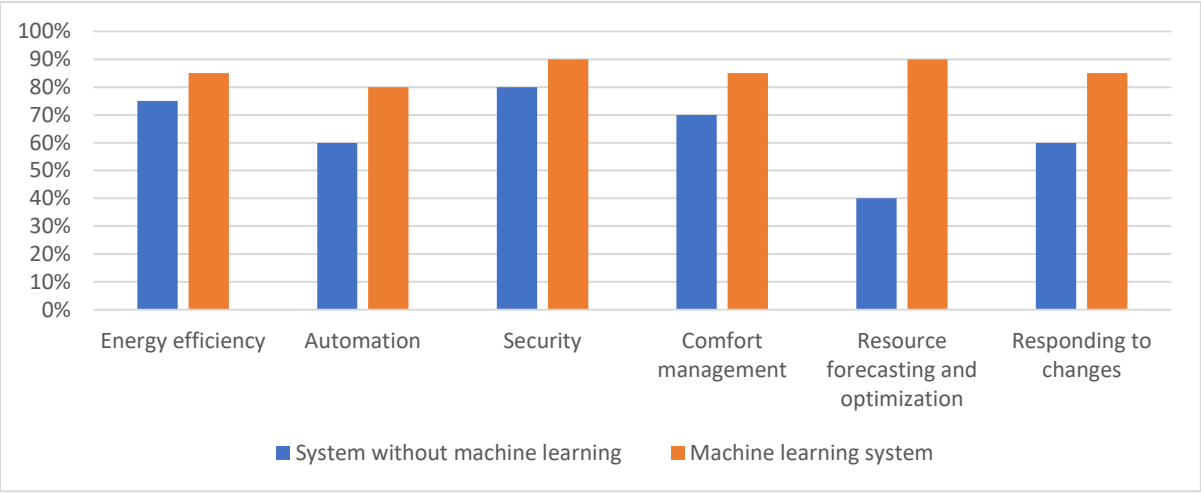

Figure 2: Smart home system performance with and without machine learning methods

3. Discussion

Machine learning helps to avoid certain problems by analyzing previous data and recognizing patterns, but it cannot predict absolutely all possible scenarios, especially if they arise from certain unpredictable factors or third-party interventions.

A smart home system that uses machine learning methods proves to be better than a system without this technology (as the study results show, the performance of a smart home using failure prediction methods gives an average of 22% better result compared to a similar system without prediction—Fig. 3). Machine learning allows the system to adapt to changes in the environment and user requirements, respond more quickly to new conditions, and optimize resource use. This helps to improve the system's efficiency in managing energy, comfort, safety, and user satisfaction [20].

Machine learning allows the system to predict and avoid failures, which ensures greater reliability and durability of the system. This approach also allows for increased automation, helping the system perform routine tasks without user intervention [21, 22]. Overall, a machine learning system remains the preferred choice due to its ability to predict, optimize, and adapt to changes, enabling it to provide more efficient and convenient smart home management [23, 24].

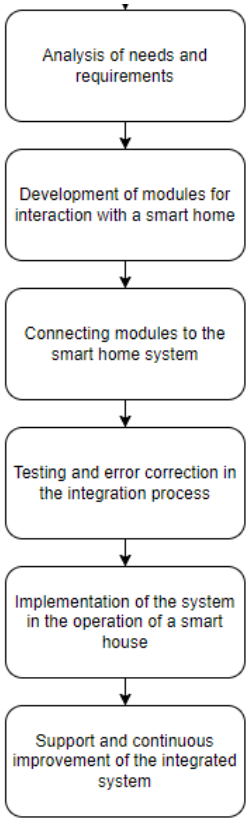

Figure 3: The process of integrating an information system into a smart home

4. Conclusions

The study results have shown a wide range of modern technologies, sensors, and control systems used in smart homes. The overview has shown that existing technologies have the potential to improve convenience, security, and energy efficiency.

The analysis of available machine learning methods indicates their potential in predicting

and managing risks in smart homes. The considered models have shown high accuracy in predicting failures.

The following areas can be considered for further development of this work:

- Improving machine learning methods to increase the accuracy of failure prediction.
- In-depth study of the impact of the introduction of machine learning systems on the functioning of a smart home.

Based on the obtained data, it is possible to build a methodology for predicting failures in a smart home, which will be the direction of the authors' next research.

References

[1] Z. Hu, et al., Bandwidth Research of Wireless IoT Switches, in: IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (2020). doi: 10.1109/tcset49122.2020.2354922

[2] I. Kuzminykh, et al., Investigation of the IoT Device Lifetime with Secure Data Transmission, Internet of Things, Smart Spaces, and Next Generation Networks and Systems, vol. 11660 (2019) 16–27. doi: 10.1007/978-3-030-30859-9_2

[3] B. Zhurakovskyi, et al., Secured Remote Update Protocol in IoT Data Exchange System, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, vol. 3421 (2023) 67–76

[4] O. Shevchenko, et al., Methods of the Objects Identification and Recognition Research in the Networks with the IoT Concept Support, in: Cybersecurity Providing in Information and Telecommunication Systems, vol. 2923 (2021) 277–282.

[5] M. Moshenchenko, et. al., Optimization Algorithms of Smart City Wireless Sensor Network Control, in: Cybersecurity Providing in Information and Telecommunication Systems II, vol. 3188 (2021) 32–42.

[6] V. Sokolov, et al., Method for Increasing the Various Sources Data Consistency for IoT Sensors, in: IEEE 9th International Conference on Problems of Info-communications, Science and Technology (2023) 522–526. doi: 10.1109/PICST57299.2022.10238518.

[7] Smart Home Technology: How AI Creates a Space that is Comfortable for life. URL: https://www.everest.ua/tehnologiya-rozumnogo-budynku-yak-ai-stvoryuye-prostirkomfortnyj-dlya-zhyttya/

[8] T. Arsan, Smart Systems: From Design to Implementation of Embedded SMART Systems (2016). doi: 10.1109/HONET. 2016.7753420.

[9] V. Zhebka et al., Optimization of Machine Learning Method to Improve the Management Efficiency of Heterogeneous Telecommunication Network, Cybersecurity Providing in Information and Telecommunication Systems Vol. 3288 (2022) 149–155.

[10] A. Boiko, Modeling of the Automated System of Operational Control of the Parameters of the "Smart Home" in the Proteus Environment, Technologies and Design 2(35) (2020).

[11] V. Zhebka, Y. Bazak, K. Storchak, Features of Predicting Failures in a Smart Home based on Machine Learning Methods, Telecommunications and Information Technologies 4(81) (2023) 4–12.

[12] I. Ruban, V. Martovytskyi, S. Partyka, Classification of Methods for Detecting Anomalies in Information Systems, Systems of Armament and Military Equipment 3(47) (2016) 47.

[13] V. Kharchenko, Fundamentals of Machine Learning, Sumy State University (2023).

[14] X. Sun, et al., System-Level Hardware Failure Prediction using Deep Learning, 56th Annual Design Automation Conference 20 (2019) 1–6. doi: 10.1145/3316781.3317918.

[15] V. Malinov, et al., Biomining as an Effective Mechanism for Utilizing the Bioenergy Potential of Processing Enterprises in the Agricultural Sector, in: Cybersecurity Providing in Information and Telecommunication Systems Vol. 3421 (2023) 223–230.

[16] Y. Bazak, Comparison of Machine Learning Methods for Predicting Failures in a Smart Home, Abstracts of the Report at the Scientific and Practical

Conference “Problems of Computer Engineering,” Collection of abstracts (2023) 125–127.
[17] K. Kononova, Machine Learning: Methods and Models: a Textbook for Bachelors, Masters and Doctors of Philosophy in Specialty 051 “Economics,” V. N. Karazin Kharkiv National University (2020).
[18] Y. Bondarenko, Manual for the Study of the Discipline “Statistical Analysis of Data,” Lira (2018).
[19] K. Hureeva, O. Kudin, A. Lisnyak, A Review of Machine Learning Methods in the Task of forecasting Financial Time Series, Computer Science and Applied Mathematics (2) (2018) 18–28.
[20] I. Puleko, A. Yefimenko, Architecture and Technologies of the Internet of Things: a Textbookб State University “Zhytomyr Polytechnic” (2022).
[21] B. Zhurakovskyi, et al., Coding for Information Systems Security and Viability, in: Information Technologies and Security Vol. 2859 (2021) 71–84.
[22] M. Moshenchenko, et al., Optimization Algorithms of Smart City Wireless Sensor Network Control, in: Cybersecurity Providing in Information and Telecommunication Systems Vol. 3188 (2021) 32–42.
[23] B. Chen, et al., Smart Factory of Industry 4.0: Key Technologies, Application Case, and Challenges, IEEE Access 6 (2018) 6505–6519. doi: 10.1109/ACCESS.2017.2783682.
[24] J. Mineraud, et al., A Gap Analysis of Internet-of-Things Platforms, Computer Communications 89–90 (2016) 5–16. doi: 10.1016/j.comcom.2016.03.015.