Сучасний стан наукових досліджень та технологій в промисловості. 2023. № 1 (23)
ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

UDC 519.6:681.518.2:004.62:519.23
DOI: https://doi.org/10.30837/ITSSI.2023.23.123

D. Shevchenko, M. Ugryumov, S. Artiukh

MONITORING DATA AGGREGATION OF DYNAMIC SYSTEMS USING INFORMATION TECHNOLOGIES

The subject matter of the article is models, methods and information technologies of monitoring data aggregation. The **goal** of the article is to determine the best deep learning model for reducing the dimensionality of dynamic systems monitoring data. The following **tasks** were solved: analysis of existing dimensionality reduction approaches, description of the general architecture of vanilla and variational autoencoders, development of their architecture, development of software for training and testing of autoencoders, conducting research on the performance quality of autoencoders for the problem of dimensionality reduction. The following models and **methods** were used: data processing and preparation, data dimensionality reduction. The software was developed using the Python language. Scikit-learn, Pandas, PyTorch, NumPy, argparse and others were used as auxiliary libraries. Obtained **results**: the work presents a classification of models and methods for dimensionality reduction, general reviews of vanilla and variational autoencoders, which include a description of the models, their properties, loss functions and their application to the problem of dimensionality reduction. Custom autoencoder architectures were also created, including visual representations of the autoencoder architecture and descriptions of each component. The software for training and testing autoencoders was developed, the dynamic system monitoring data set, and the steps for pre-training the data set were described. The metric for evaluating the quality of models is also described; the configuration of autoencoders and their training are considered. **Conclusions**: The vanilla autoencoder recovers the data much better than the variational one. Looking at the fact that the architectures of the autoencoders are the same, except for the peculiarities of the autoencoders, it can be noted that a vanilla autoencoder compresses data better by keeping more useful variables for later recovery from the bottleneck. Additionally, by training on different bottleneck sizes, you can determine the size at which the data is recovered best, which means that the most important variables are preserved. Looking at the results in general, the autoencoders work effectively for the dimensionality reduction task and the data recovery quality metric shows that they recover the data well with an error of 3–4 digits after 0. In conclusion, the vanilla autoencoder is the best deep learning model for aggregating monitoring data of dynamic systems.

Keywords: data dimensionality reduction; deep learning; autoencoders.

Problem statement and its relevance

A dynamic system is one where the function describes the time dependence of a point in the surrounding space. An example of such a system is an economic system based on monitoring data, where there is a time dependence of system variables (days, months, years, etc.). The results of monitoring can be:

– data samples. These are sets of values for a certain period of time, which can vary significantly depending on the conditions of the system;

– time series. They are presented as sets of measurements of a variable that are closely related to each other and obtained within a certain period of time during which the values of the variable do not change significantly. Time series are discrete models for monitoring the state of a dynamic system, which usually contain parametric uncertainties, are non-stationary and noisy.

In order to solve the problem of dimensionality reduction, it is necessary to define the basic provisions on this issue. Dimensionality reduction is the transformation of data from a high-dimensional space to a low-dimensional space in such a way that the low-dimensional representation preserves some significant properties of the original data, ideally close to its intrinsic dimensionality. Working in high-dimensional spaces can be undesirable for many reasons: raw data is often sparse due to the curse of dimensionality, and data analyses are usually hard to compute (difficult to control or work with). The main advantages of using dimensionality reduction methods are:

– removal of variables that do not have important information;

– reduction of multicollinearity in the data;

– reduction of the required data storage space;

– reducing the time required for data-related calculations;

– presenting the data in a way that allows for visualisation.

Usually, it is necessary to preprocess the data in time series and data samples by removing missing values and preparing the data for submission to dimensionality reduction techniques. Once the data has been preprocessed, the dimensionality reduction techniques can be used.

The task of reducing the dimensionality of monitoring data in dynamic systems should be defined as a sequential performance of interrelated tasks, namely:

– monitoring the state of a dynamic system;

– pre-processing of the data, which makes it possible to give them a certain form that is allowed for the use of dimensionality reduction methods;

– reduction of the dimensionality of the monitoring data of a dynamic system.

Considering the theoretical and practical problems of the dimensionality reduction of monitoring data of dynamic systems is of great interest to scientists in Ukraine and abroad. To date, a number of papers have been published describing models and methods for dimensionality reduction [1–15]. V. Hrusha's work [1] discusses basic methods for dimensionality reduction, such as: the use of geometric parameters of the FEM; selection of FEM values on a nonlinear scale; use of coefficients of approximating polynomials; application of the principal components method. In their studies, V. Martsenyuk, Y. Droniak, I. Tsikorska [2] and E. Kozak [3] rely on the principal component method to solve data aggregation problems. Paper [4] outlines the theoretical and practical foundations of using a deep learning model to predict monitoring data of dynamic systems using data aggregation modules. Researchers M. Korabliov and S. Lutskyi in [5] described the main methods for information processing, where they identified the task of dimensionality reduction as one of the important components of processing. The existing methods and models for dimensionality reduction are considered in [6–8, 11, 15]. The authors theoretically described various methods and models without practical support. It is also worth mentioning [9, 14], where scientists developed new deep learning models for dimensionality reduction based on supervised learning. The use of autoencoders for unsupervised learning is discussed in [10], where an autoencoder is defined as the main model for dimensionality reduction and data recovery. It is also worth mentioning foreign works by the following authors: P. May, H. Rekabdarkolaee [12], K. Matchev, K. Matcheva, A. Roman [13] and others.

There are two basic approaches to dimensionality reduction: *Feature Selection* and *Dimensionality Reduction*. The feature selection approach is based on the fact that the most important variables are selected, thus reducing the dimensionality of the data. In addition, the methods of the dimensionality reduction approach perform certain transformations in order to obtain data of lower dimensionality. The classification of dimensionality reduction models and methods is shown in Fig. 1.

The classification offers most of the popular models and methods for dimensionality reduction, which are actively used depending on the requirements and goals. In this paper, we will consider deep learning models [16] in more detail, namely the autoencoder and the variational autoencoder.

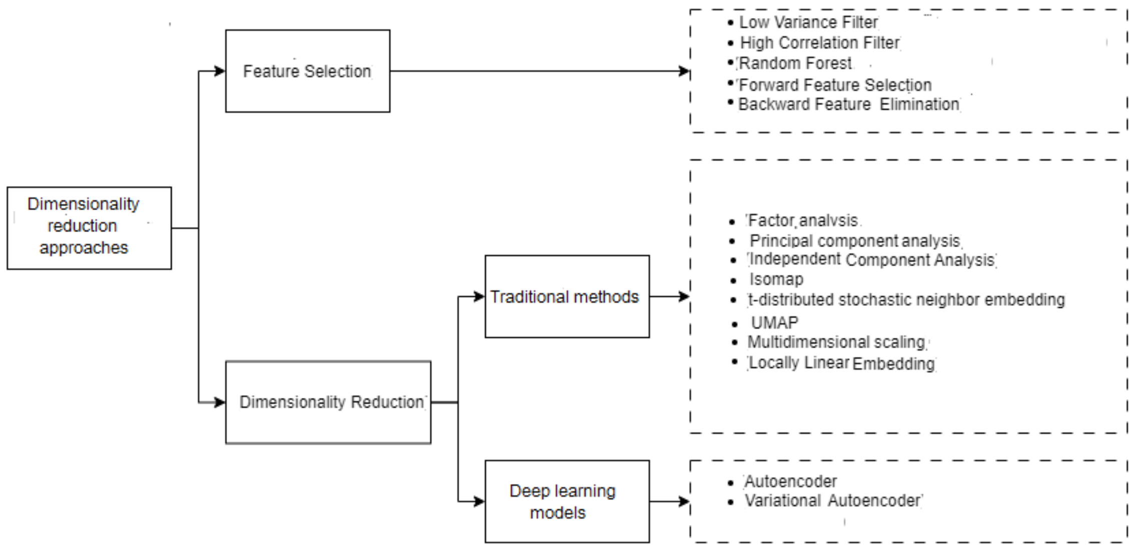

Dimensionality reduction approaches
Feature Selection
Dimensionality Reduction
Traditional methods
Deep learning models
• Low Variance Filter
• High Correlation Filter
• Random Forest
• Forward Feature Selection
• Backward Feature Elimination
• Factor analysis
• Principal component analysis
• Independent Component Analysis
• Isomap
• t-distributed stochastic neighbor embedding
• UMAP
• Multidimensional scaling
• Locally Linear Embedding
• Autoencoder
• Variational Autoencoder

Fig. 1. Classification of models and methods of dimensionality reduction

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

Having analysed the works devoted to solving dimensionality reduction problems, it should be emphasised that they do not determine the best deep learning model, which are shown in Fig. It is also worth noting that the information support developed to date does not allow solving the problem of dimensionality reduction in data samples and time series with a high level of reliability. Therefore, there is a need to choose the best deep learning model and information technology implementation tools to reduce the dimensionality of monitoring data, which determines the relevance of the study.

The purpose of the study is to determine the best deep learning model for reducing the dimensionality of monitoring data of dynamic systems. The object is dynamic systems, and the subject is models, methods and information technologies for aggregating monitoring data.

1 Standard autoencoder

A standard autoencoder can be described as a type of feed-forward neural network where the input is the same as the output. It compresses the input data into a bottleneck (lower dimensional data) and then reconstructs the output data from this representation. The bottleneck is a compact summation or compression of the input data, also called a latent space representation.

An autoencoder has three components: an encoder, a bottleneck, and a decoder. The encoder compresses the input data and creates a narrow throat, the decoder then reconstructs the input data using only this throat. The architecture of the autoencoder is shown in Fig. 2, which shows all the main components of the neural network.

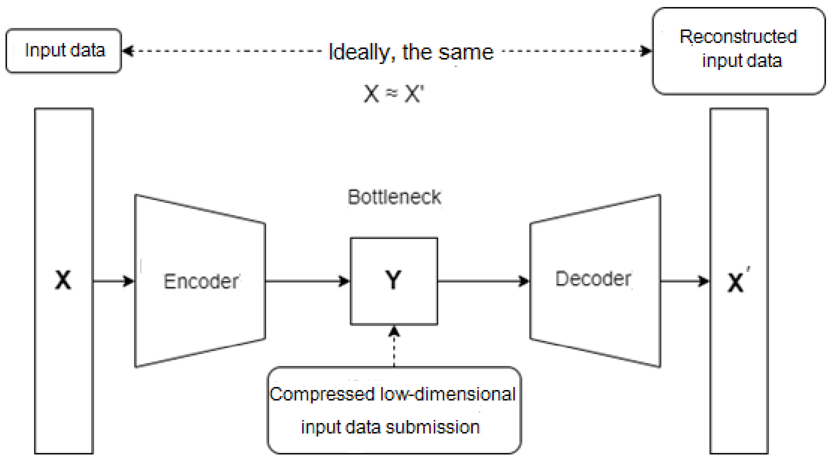


Fig. 2. General architecture of a standard autoencoder

An autoencoder is basically a dimensionality reduction (or aggregation) algorithm with several important properties:

– data dependence. An autoencoder is only able to compress the data significantly, similar to what it has been trained on. Since it learns functions specific to the given training data, it differs from a standard data compression algorithm. Therefore, it should not be used on data that differs from the training distribution;

– there are losses. The output of the autoencoder (or the reconstructed input data) will not be the same as the input data. It will be a close but degraded representation;

– learning without a teacher. The autoencoder belongs to mathematical models with unsupervised learning techniques, as it does not need explicit labels to learn, but only needs to be given input. However, to be more precise, the autoencoder is self-supervised, as it generates its own labels from the training data. The model is trained by minimising a chosen loss function that explores the error between the input and reconstructed data.

Considering an autoencoder for the task of data dimensionality reduction, it can be determined that the data generated in the bottleneck is the target data. That is, to use an autoencoder for dimensionality reduction, it is necessary to train it on the input data and apply the encoder to obtain dimensionality-reduced data.

2 Variational autoencoder

A variational autoencoder is an extension of the above autoencoder. The variational autoencoder shares properties with the standard autoencoder, including data dependence, loss, and unsupervised learning.

The difference is that the variational autoencoder provides a probabilistic way to describe an observation in a bottleneck (latent space). So, instead of designing an encoder that outputs a single value to describe each attribute of the latent state, an encoder is created to describe the probability distribution for each latent attribute.

The architecture of the variational autoencoder (see Fig. 3) is similar to that of the standard one, except for the bottleneck. Now, the encoder extracts two values, such as the mean and variance, which are used in the subsequent formation of the bottleneck.

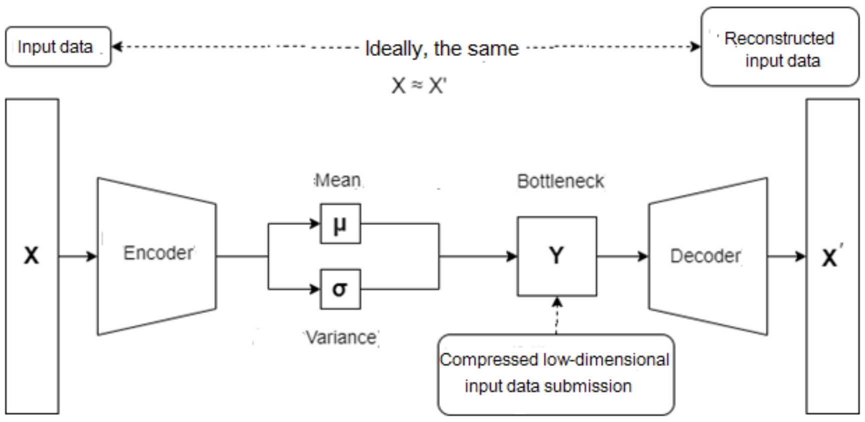


Fig. 3. General architecture of a variable autoencoder

Considering the process of training such a mathematical model, it can be noted that the loss function used is different from the one used in the autoencoder. Instead of the root mean square error, the loss function is used, defined as follows:

$$L = MSE\left(X, X^{'}\right) + KL\left(\mu, \sigma\right), \quad (1)$$

where MSE – standard error; KL – Kulbak–Leibler divergence.

If we consider a variational autoencoder for the dimensionality reduction task, then, similarly to an autoencoder, the data generated in the narrow throat is targeted. This means that you need to train the variational autoencoder on the input data and take the data from the bottleneck in order to obtain the reduced dimensionality data.

3 Description of autoencoder architecture

There are many variations in the architecture of autoencoders depending on the type and complexity of the task. If you use an autoencoder to reduce the dimensionality of tabular data, you should use a conventional feed-forward neural network. In this case, it makes no sense to use convolutional neural networks, as they are usually required for working with images.

Both types of autoencoders have been designed with the same architecture, except for the difference between a standard and a variation autoencoder. The auto-encoders have three linear layers in the encoder, one layer in the narrow throat and three linear layers in the decoder. The same architecture was chosen in order to determine the best model for dimensionality reduction with high accuracy, based on studies where the difference in their results depends only on the specific differences that characterise standard and variation autoencoders.

3.1 Architecture of a standard autoencoder

The architecture of the standard autoencoder that will be used in this paper is shown in Fig. 4. The diagram clearly identifies three parts of the autoencoder: the encoder, the bottleneck, and the decoder. The labels above the arrows indicate the size of the data fed to the next layer of the neural network. The size is determined by variables that can be controlled, namely:

– *batch_size* characterises the size of the data packet fed to the neural network;

– *input_dim* defines the size of input state variables;

– *h1*, *h2* are the sizes of the hidden layers of the neural network;

– *latent_dim* – is the size of the data in the bottleneck. This variable determines the dimensionality of the data that will be obtained after dimensionality reduction.

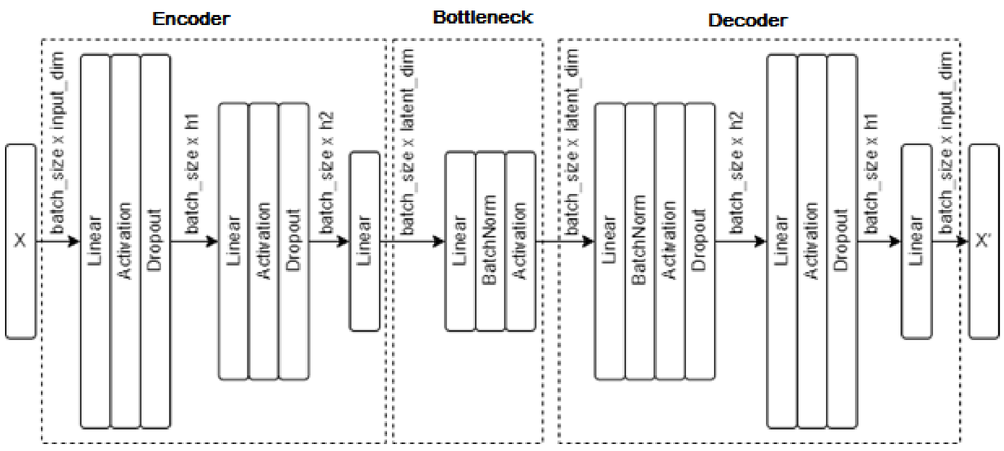


Fig. 4. Autoencoder architecture

Also in Fig. 4, the variables characterising the types of neural network layers are defined, in particular:

– *Linear*. This layer uses a linear transformation to the input data: $y = x \times A^T + b$.

– *Activation*. Uses the activation function before the input data.

– *Dropout*. During training, the layer randomly zeroes some of the input data elements with a certain probability using samples from the Bernoulli distribution. It has been proven as an effective method of regularization and prevention of neuronal coadaptation, as described in [17].

– *BatchNorm*. Applies batch normalization to input data with dimension 2 (for example, data with dimension 16 x 128), as described in [18].

3.2 Architecture of the variational autoencoder

The architecture of the variational autoencoder that will be used in this paper is shown in Fig. 5. The diagram identifies four main parts: the encoder, μ and σ , the narrow throat, and the decoder. The labels above the arrows indicate the data size, similar to the diagram of a standard autoencoder. Their definitions are described in p. 3.1.

The diagram shows the variables characterizing the types of neural network layers. They are similar to those described in section 3.1. The variational autoencoder provides distribution parameters (μ and σ), which are then used to generate data in the narrow throat.

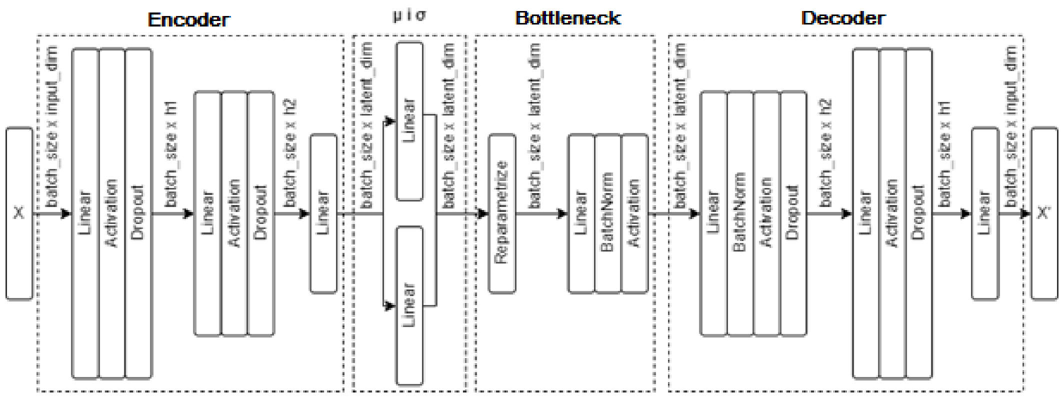


Fig. 5. Architecture of the variational autoencoder

This process of sampling from the distribution parameterized by the model cannot be differentiated. Therefore, an additional layer type was added – *Reparametrize*. It allows you to build a gradient path through a non-stochastic node and further differentiate this node. The main idea of this layer is to add noise, which is obtained from the normal distribution, to μ and σ .

This layer can be described by the following formula:

$$y = \mu + \sigma \times \epsilon , \qquad (2)$$

where μ is the mean; σ is the variance; ϵ is the noise obtained randomly from a normal distribution. Thus, the random element was separated from the studied parameterization, and now it is possible to differentiate the model completely.

ISSN 2522-9818 (print)
ISSN 2524-2296 (online) Innovative technologies and scientific solutions for industries. 2023. No. 1 (23)

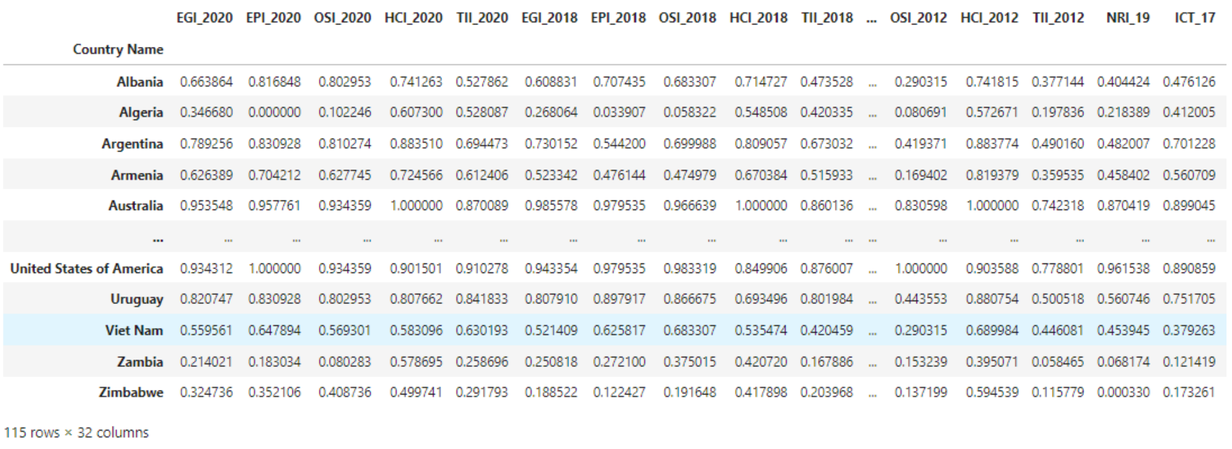
4 Overview of model results and performance

4.1 Data set

To train and test the auto-coders, we use a proprietary dataset that reproduces monitoring data in an economic dynamic system. It reproduces the socio-economic development of countries for 2012–2020. The set consists of 115 alternatives with 32 state variables.

The overview part of the dataset is shown in Fig. 6.

This dataset was preprocessed by removing/replacing missing values, applying min-max normalization in the range from 0 to 1 inclusive, and given a form that is acceptable for training and testing of autoencoders. Moreover, the dataset is divided into two parts to be used for training and testing, where 70% of all data relates to the training part and the rest to the testing part.



	EGI_2020	EPI_2020	OSI_2020	HCI_2020	TII_2020	EGI_2018	EPI_2018	OSI_2018	HCI_2018	TII_2018	...	OSI_2012	HCI_2012	TII_2012	NRI_19	ICT_17
Country Name																
Albania	0.663864	0.816848	0.802953	0.741263	0.527862	0.608831	0.707435	0.683307	0.714727	0.473528	...	0.290315	0.741815	0.377144	0.404424	0.476126
Algeria	0.346680	0.000000	0.102246	0.607300	0.528087	0.268064	0.033907	0.058322	0.548508	0.420335	...	0.080691	0.572671	0.197836	0.218389	0.412005
Argentina	0.789256	0.830928	0.810274	0.883510	0.694473	0.730152	0.544200	0.699988	0.809057	0.673032	...	0.419371	0.883774	0.490160	0.482007	0.701228
Armenia	0.626389	0.704212	0.627745	0.724566	0.612406	0.523342	0.476144	0.474979	0.670384	0.515933	...	0.169402	0.819379	0.359535	0.458402	0.560709
Australia	0.953548	0.957761	0.934359	1.000000	0.870089	0.985578	0.979535	0.966639	1.000000	0.860136	...	0.830598	1.000000	0.742318	0.870419	0.899045
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
United States of America	0.934312	1.000000	0.934359	0.901501	0.910278	0.943354	0.979535	0.983319	0.849906	0.876007	...	1.000000	0.903588	0.778801	0.961538	0.890859
Uruguay	0.820747	0.830928	0.802953	0.807662	0.841833	0.807910	0.897917	0.866675	0.693496	0.801984	...	0.443553	0.880754	0.500518	0.560746	0.751705
Viet Nam	0.559561	0.647894	0.569301	0.583096	0.630193	0.521409	0.625817	0.683307	0.535474	0.420459	...	0.290315	0.689984	0.446081	0.453945	0.379263
Zambia	0.214021	0.183034	0.080283	0.578695	0.258696	0.250818	0.272100	0.375015	0.420720	0.167886	...	0.153239	0.395071	0.058465	0.068174	0.121419
Zimbabwe	0.324736	0.352106	0.408736	0.499741	0.291793	0.188522	0.122427	0.191648	0.417898	0.203968	...	0.137199	0.594539	0.115779	0.000330	0.173261

115 rows × 32 columns

Fig. 6. Data set

4.2 Quality assessment metrics

In order to determine the best model for dimensionality reduction, the root mean square error is used as a metric to evaluate the quality of the model. This metric is calculated by the following formula:

$$MSE\left(X, X'\right) = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - X_i'\right)^2, \qquad (3)$$

where, in the case of autoencoders, X is the input data; X' is the reconstructed input data. There is an inverse relationship between the metric and the quality of the model. That is, the lower is the metric value, the higher is the quality of the model, and it reduces the data dimensionality better, preserving more useful variables.

4.3 Autoencoders configuration and training

Part 3 describes the architectures of autoencoders and introduces variables that control the sizes of data and the corresponding layers of the neural network. Also, such layers as *Dropout* and *Activation* have their own parameters. All the values of variables and parameters are shown in Table 1.

An optimization algorithm called *Adam* [19] was used to train autoencoders.

Standard parameters for this algorithm were used, namely:

– learning rate $= 1e-3$;

– coefficient used to calculate the gradient moving averages = 0.9;

– the coefficient used to calculate the square of the gradient = 0.999.

Table 1. *Configuration of autoencoders*

Variable, parameter	Value
batch_size	16
input_dim	32
h1	128
h2	64
latent_dim	20
Probability with a layer *Dropout*	0.2
Activation function in the layer *Activation*	ReLU

The loss functions used in training the autoencoders are also defined. They are different for the standard and variational autoencoders. Thus, the MSE loss function (3) is used for the autoencoder, and the function defined in (1), which is a combination of MSE and Kulbak-Leibler divergence, is used for the variational autoencoder.

4.4 Software

The software was developed using the Python language. The following libraries were used as auxiliary libraries:

– *scikit-learn* for applying normalisation, quickly dividing the data set into training and testing parts;

– *Pandas* for working with data, processing it, etc.

– *PyTorch* for working with neural networks, training and testing them, creating loss functions, etc.

– *NumPy* for working with matrix calculations and initialising model parameters;

– *argparse* for working with console parameters

– and others, which are insignificant for their description.

As a result, a console application was developed that allows you to perform the following actions:

– loading a tabular dataset, extracting the specified state variables, and splitting it into training and testing parts;

– training of standard and variation autoencoders using configuration parameters;

– saving the best version of the autoencoder;

– continue training the model with the existing version of the model;

– calculation of performance indicators and loss functions;

– testing standard and variation autoencoders using configuration parameters;

– use of trained autoencoders with four options: encoding, encoding + sampling, decoding, and full model pass.

The software supports the execution of the program on the central processing unit (CPU) and the graphics processing unit (GPU). It should be noted that the use of the GPU significantly increases the speed of training and testing of autoencoders.

4.5 Results of training and testing of autoencoders

The training of the autoencoders was carried out with a fixed number of epochs, namely 1000 epochs for each training. After every ten epochs, the models were tested using the metric described in Section 4.2. Based on the test results, the best version of the model was selected after the training. Additionally, the training was performed by changing the *latent_dim* parameter, which is responsible for the dimensionality of the data placed in the bottleneck, i.e. the target data for the dimensionality reduction task. The results of testing the autoencoders are presented in Table 2.

Based on the results of testing the autoencoders, it can be determined that the standard autoencoder recovers data much better than the variational one. Given that the architectures of the autoencoders are identical, except for the features of the autoencoders, it can be noted that the standard autoencoder compresses data better, preserving more useful variables for further recovery from the bottleneck. Also, by training on different sizes of the bottleneck, you can determine the most effective size at which the data is recovered best, which means that the most important variables are preserved. In this case, if the size is 10, both autoencoders work most efficiently.

Table 2. *Results of autoencoders testing*

Autoencoder type	latent_dim	MSE
Standard	20	7.1×10^{-4}
Variational		16.7×10^{-4}
Standard	15	6.8×10^{-4}
Variational		15.8×10^{-4}
Standard	10	**6.6×10^{-4}**
Variational		14.8×10^{-4}
Standard	5	7.6×10^{-4}
Variational		15.2×10^{-4}
Standard	2	12.6×10^{-4}
Variational		15.4×10^{-4}
Standard	1	10.9×10^{-4}
Variational		14×10^{-4}

It should be noted that when aggregating to dimension 1, the variational autoencoder recovers the data best. This phenomenon encourages future research. On the other hand, it is a good reason to reduce the dimensionality for data visualisation.

Accordingly, general autoencoders work well for the dimensionality reduction task, and the data recovery quality metric shows that they recover data well with an error of 3–4 digits after 0.

Results and conclusions

The paper presents a classification of models and methods for reducing the dimensionality of monitoring data of dynamic systems, which involves the division into two approaches, such as variable selection and dimensionality reduction. Considering deep learning models in relation to the dimensionality reduction approach, standard and variational autoencoders are selected.

A general overview of the selected autoencoders is offered, including a description of the models, their properties, loss functions, and their application to the task of data dimensionality reduction. We also created our own autoencoder architectures, described in Section 3, including a visual representation of the autoencoder architecture and a description of each component.

To train and test the autoencoders, we developed software using the *Python* language and auxiliary libraries (see p. 4.4). We also present the monitoring data set of the dynamic system used in this work. Additionally, the steps for preliminary preparation of the dataset to give it a form that can be used in the developed software are described. In the course of the study of training and testing of autoencoders, a metric for evaluating the quality of models is described, the configuration of autoencoders and their training is considered, the sizes of autoencoder layers, parameters of the optimisation algorithm, loss functions, etc. are given.

As a result of training and testing the autoencoders, the best values of the metric were obtained for different sizes of the narrow throat, which characterises data of reduced dimensionality. A total of 12 autoencoders were trained and tested, half of which were variational. After testing them, conclusions were drawn on the results and it was determined that the standard autoencoder performs better on all narrow-neck sizes that were investigated. It should be noted that the variational autoencoder requires more training time and converges more slowly.

Therefore, the standard autoencoder is the best deep learning model for reducing the dimensionality of dynamic system monitoring data. In further research, a model such as the standard autoencoder can be used to recover monitoring data in automated process control systems, reduce data dimensionality in machine learning using *Big Data* technologies.

References

1. Grusha, V. M. (2017), "Chlorophyll fluorometer data normalization and dimensionality reduction” ["Normalizaciya ta zmenshennya rozmirnosti dany’x xlorofil-fluorometriv"], *Computer facilities, networks and systems*, No. 16, P. 76–86.
2. Martseniuk, V. P., Dronyak, Y. V. and Tsikorska, I. V. (2019) "Reduction of dimension for prediction of progress in problems of medical education: an approach based", *Medical Informatics and Engineering*, No. 4, P. 16–24.
3. Kozak, Ye. B. (2021), "A complex algorithm for creating control automata based on machine learning" ["Kompleksny’j algory`tm stvorennya keruyuchy’x avtomativ na bazi mashy`nnogo navchannya"], *Technical engineering*, No. 2 (88), P. 35–41. DOI: https://doi.org/10.26642/ten-2021-2(88)-35-41.
4. Bakurova, A. et al. (2021), "Neural network forecasting of energy consumption of a metallurgical enterprise", Innovative *Technologies and Scientific Solutions for Industries*, No. 1 (15), P. 14–22. DOI: https://doi.org/10.30837/itssi.2021.15.014.
5. Korablyov, M. and Lutskyy, S. (2022), "System-information models for Intelligent Information Processing", *Innovative Technologies and Scientific Solutions for Industries*, No. 3 (21), P. 26–38. DOI: https://doi.org/10.30837/itssi.2022.21.026.
6. Xie, H., Li, J. and Xue, H. (2018), "A survey of dimensionality reduction techniques based on random projection", *arXiv.org*. DOI: https://doi.org/10.48550/arXiv.1706.04371
7. Espadoto, M. et al. (2021), "Toward a quantitative survey of dimension reduction techniques," *IEEE Transactions on Visualization and Computer Graphics*, No. 27 (3), P. 2153–2173. DOI: https://doi.org/10.1109/tvcg.2019.2944182
8. Velliangiri, S., Alagumuthukrishnan, S. and Thankumar joseph, S.I. (2019), "A review of dimensionality reduction techniques for efficient computation", *Procedia Computer Science*, No. 165, P. 104–111. DOI: https://doi.org/10.1016/j.procs.2020.01.079
9. McInnes, L., Healy, J. and Melville, J. (2020), "UMAP: Uniform manifold approximation and projection for dimension reduction", *arXiv.org*. DOI: https://doi.org/10.48550/arXiv.1802.03426
10. Chorowski, J. et al. (2019), "Unsupervised speech representation learning using WaveNet autoencoders", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, No. 27 (12), P. 2041–2053. DOI: https://doi.org/10.1109/TASLP.2019.2938863
11. Jia, W. et al. (2022), "Feature dimensionality reduction: A Review", Complex &amp; *Intelligent Systems*, No. 8 (3), P. 2663–2693. DOI: https://doi.org/10.1007/s40747-021-00637-x
12. May, P. and Rekabdarkolaee, H.M. (2022), "Dimension reduction for spatially correlated data: Spatial predictor envelope", *arXiv.org*. DOI: https://doi.org/10.48550/arXiv.2201.01919
13. Matchev, K.T., Matcheva, K. and Roman, A. (2022), "Unsupervised machine learning for exploratory data analysis of Exoplanet Transmission Spectra", *arXiv.org*. DOI: https://doi.org/10.48550/arXiv.2201.02696
14. Björklund, A., Mäkelä, J. and Puolamäki, K. (2022), "SLISEMAP: Supervised dimensionality reduction through local explanations", *Machine Learning*, No. 112 (1), P. 1–43. DOI: https://doi.org/10.1007/s10994-022-06261-1
15. Bhandari, N. et al. (2022), "A comprehensive survey on computational learning methods for analysis of Gene Expression Data", *arXiv.org*. DOI: https://doi.org/10.48550/arXiv.2202.02958
16. Bank, D., Koenigstein, N. and Giryes, R. (2021), "Autoencoders", *arXiv.org*. DOI: https://doi.org/10.48550/arXiv.2003.05991
17. Hinton, G.E. et al. (2012), "Improving neural networks by preventing co-adaptation of feature detectors", *arXiv.org*. DOI: https://doi.org/10.48550/arXiv.1207.0580

ISSN 2522-9818 (print)
ISSN 2524-2296 (online)

18. Ioffe, S. and Szegedy, C. (2015), "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *International conference on machine learning*. DOI: https://doi.org/10.48550/arXiv.1502.03167
19. Kingma, D.P. and Ba, J. (2017), "Adam: A method for stochastic optimization", *arXiv.org*. DOI: https://doi.org/10.48550/arXiv.1412.6980

Received 31.03.2023

Відомості про авторів / About the Authors

Шевченко Дмитро Олександрович – аспірант, Харківський національний університет ім. В. Н. Каразіна, Харків, Україна; e-mail: dimyich24@gmail.com; ORCID ID: https://orcid.org/0000-0003-0902-2735

Угрюмов Михайло Леонідович – доктор технічних наук, Харківський національний університет ім. В. Н. Каразіна, професор кафедри теоретичної та прикладної системотехніки; Харків, Україна; e-mail: ugryumov.mykhaylo52@gmail.com; ORCID ID: https://orcid.org/0000-0003-0902-2735

Артюх Сергій Володимирович – кандидат медичних наук, Державна установа "Інститут медичної радіології та онкології ім. С. П. Григор'єва Національної академії медичних наук України", старший науковий співробітник; Харківський національний медичний університет, асистент кафедри; e-mail: artiukhsergii@ukr,net; ORCID ID: https://orcid.org/0000-0002-7189-3614

Shevchenko Dmytro – V. N. Karazin Kharkiv National University, Postgraduate Student, Kharkiv, Ukraine.

Ugryumov Mykhaylo – doctor of Engineering Sciences, V. N. Karazin Kharkiv National University, professor of Theoretical and Applied Systems Engineering Department, Kharkiv, Ukraine.

Artiukh Sergii – Candidate of Medical Sciences (PhD), State Organization "Grigoriev Institute for Medical Radiology and Oncology of the National Academy of Medical Sciences of Ukraine", senior researcher; Kharkiv National Medical University, assistant of the department.

АГРЕГУВАННЯ ДАНИХ МОНІТОРИНГУ ДИНАМІЧНИХ СИСТЕМ ІЗ ВИКОРИСТАННЯМ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Предметом роботи є моделі, методи та інформаційні технології агрегування даних моніторингу. **Мета статті** – визначити найкращу модель глибокого навчання для зменшення розмірності даних моніторингу динамічних систем. **Завдання**, що вирішуються: аналіз наявних підходів зменшення розмірності, опис загальної архітектури стандартного й варіаційного автокодувальників, розроблення їх архітектури, створення програмного забезпечення для тренування й тестування автокодувальників, дослідження якості роботи автокодувальників для зменшення розмірності. Застосовано такі **методи**: підготовка та оброблення даних, зменшення розмірності даних. Програмне забезпечення було розроблено за допомогою мови *Python*. Допоміжними бібліотеками використані такі: *scikit-learn, Pandas, PyTorch, NumPy, argparse* тощо. Здобуті **результати**: у роботі запропоновано класифікацію моделей і методів для зменшення розмірності та подано загальні характеристики стандартного й варіаційного автокодувальників, що містять опис моделей, їх властивості, функції втрат та їх застосування для зменшення розмірності даних. Також створено власні архітектури автокодувальників, зокрема візуальне подання архітектури автокодувальників та опис кожного складника. Розроблено програмне забезпечення для тренування й тестування автокодувальників, розглянуто набір даних моніторингу динамічної системи та дії з попередньої підготовки набору даних. Крім того, описано метрику для оцінювання якості моделей, розглянуто конфігурацію автокодувальників та їх тренування. **Висновки**: стандартний автокодувальник відновлює дані набагато краще, ніж варіаційний. Зважаючи на те, що архітектури автокодувальників однакові, за винятком особливостей автокодувальників, можна зазначити, що стандартний автокодувальник стискає дані краще, зберігаючи більше корисних змінних для подальшого відновлення з вузького горла. Також за допомогою тренувань на різних розмірах вузького горла можна визначити розмір, за умови якого дані відновлюються найкраще, а це означає, що зберігаються найважливіші змінні. Відповідно до загальних результатів автокодувальники ефективно працюють над завданням зменшення розмірності, і метрика якості відновлення даних показує, що вони добре відновлюють дані з похибкою, яка становить 3–4 знаки після 0. Отже, стандартний автокодувальник є найкращою моделлю глибокого навчання агрегування даних моніторингу динамічних систем.

Ключові слова: зменшення розмірності даних; глибоке навчання; автокодувальники.

Бібліографічні описи / Bibliographic descriptions

Шевченко Д. О., Угрюмов М. Л., Артюх С. В. Агрегування даних моніторингу динамічних систем із використанням інформаційних технологій. *Сучасний стан наукових досліджень та технологій в промисловості*. 2023. № 1 (23). С. 123–131. DOI: https://doi.org/10.30837/ITSSI.2023.23.123

Shevchenko, D., Ugryumov, M., Artiukh, S. (2023), "Monitoring data aggregation of dynamic systems using information technologies", *Innovative Technologies and Scientific Solutions for Industries*, No. 1 (23), P. 123–131. DOI: https://doi.org/10.30837/ITSSI.2023.23.123