https://doi.org/10.33108/visnyk_tntu
Scientific Journal of the Ternopil National Technical University
2025, № 3 (119) https://doi.org/10.33108/visnyk_tntu2025.03
ISSN 2522-4433. Web:

UDC 681.3

COMPARATIVE ANALYSIS OF MLP AND KAN NEURAL NETWORK ARCHITECTURES IN NEUROINTERFACE TECHNOLOGIES

Yuriy Petrov; Oleh Pastukh

Ternopil Ivan Puluj National Technical University, Ternopil, Ukraine

This article explores the relevance of neurointerface technologies, particularly for assisting individuals with disabilities through advanced prosthetics. It examines the use of two neural network architectures, MLP (multilayer perceptron) and KAN (Kolmogorov Arnold network), for classifying finger movements based on brain signals. Results indicate that KAN models show an advantage in accuracy with smaller datasets and a more compact model size, though they require more computational resources and longer training times. In contrast, MLP is faster to train and slightly more effective on larger datasets, highlighting the potential for further development in neurointerface-based prosthetic solutions.

neural network, neurointerface, MLP, KAN, brain-computer interface, artificial intelligence, machine learning.

https://doi.org/10.33108/visnyk_tntu2025.03.107 *Received 07.08.2025*

1. INTRODUCTION

Today, it is hard to deny the relevance of neurointerface technologies. The number of people injured in disasters, wars, and accidents is growing, thus increasing the need for prosthetics. Neuro-interface prostheses are becoming increasingly important in the medical field, as they can restore lost movement capabilities and help people with disabilities return to active life. Modern neurointerface technologies allow people to control electronic devices or bionic prostheses using brain signals, thus opening up new opportunities in treatment and rehabilitation. This area is made possible by the collaboration of research institutes, private companies and universities that work intensively to improve these technologies.

With the development of computer technology, the process of implementing bionic prostheses and devices with a neural interface is growing rapidly. Research and development on this subject has been carried out by many companies and universities. Among them: The University of Michigan [1], BrainGate [2], Neuralink [3], Stanford University [4], and others [5–13].

The purpose of the article is to analyse the principle of operation, efficiency and accuracy of two neural network architectures: MLP and KAN. The comparison will be made on the basis of the results of the finger movement classification task in accordance with brain signals.

2. DATASET FORMATION

To train both neural networks, we will use data obtained experimentally using a 16-channel electroencephalograph. During the experiment, changes in brain electrical activity were recorded when a participant performed sequential finger flexion and extension movements for 2 minutes. Such actions were performed separately for the thumb and index finger, which allows us to distinguish between signals of different motor activities. The electrode placement scheme shown in the figure (Fig. 1) covers the relevant brain areas associated with the motor activity of each of the fingers, which allows obtaining data for further analysis and training of the neural network.

Keywords:

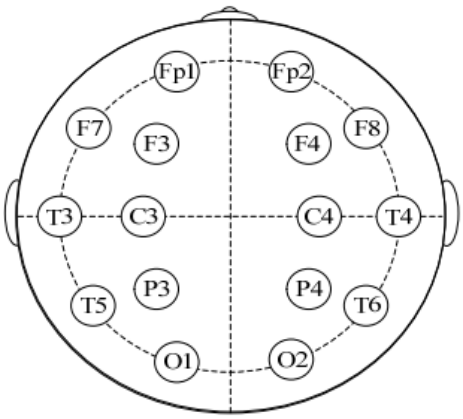

Figure 1. Diagram of electrodes on the EEG [14]

Each electrode provides a value that is the potential difference with the reference electrode attached to the earlobe. This allows us to create a data set in a table format. Here, the row is the moment of time, and the columns are the potential difference between the electrodes of the encephalograph (Fig. 2).

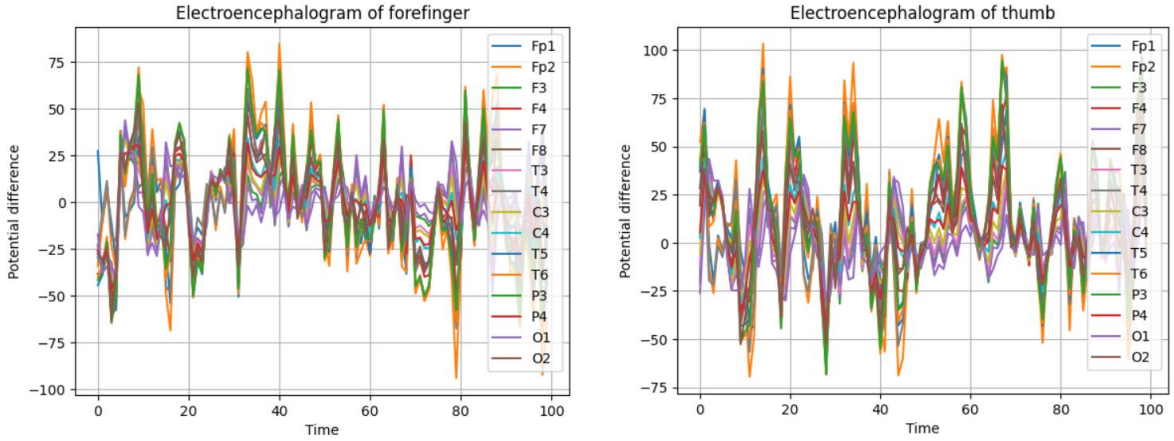

Figure 2. Graphs of electrode potential difference

3. MLP

MLP (multilayer perceptron) is a feedforward neural network architecture. An MLP unit is an artificial neuron that replicates the behaviour of a biological neuron using mathematical operations. The architecture is based on connections between neuronal layers. Each layer with the next forms a graph in which all vertices of the first layer have edges with each vertex of the second layer (Fig. 3).

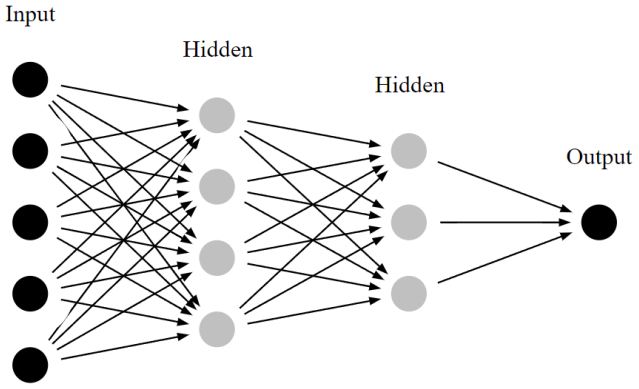

Figure 3. MLP architecture [17]

The edges contain numerical values – synaptic weights. The product of the weights together with the data sent by the neuron from the first layer is passed to the neuron in the second layer. The neuron sums up all the products it receives and passes the sum to the activation function. The result of the function is passed to all neurons in the next layer.

Using the error function, the backpropagation algorithm, and optimisation algorithms MLP performs the following steps:

1. Find out how much the network is wrong.
2. Find out how much each connection between neurons influenced the final result.
3. Find out how much the synaptic weights need to be changed to reduce the error.
4. Update the synaptic weights.

By iterating these steps, MLP «learns» and finds the dependence between input and output data.

4. KAN

The KAN architecture is a new approach to neural network architectures, which was originally described in an article by the University of Massachusetts [15], published on April 30, 2024. It is based on the Kolmogorov-Arnold representation theorem.

$$f(\mathbf{x}) = f(x_1, \dots, x_n) = \sum_{q=0}^{2n} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right) \tag{1}$$

In the context of neural networks, it is important to understand that to implement the architecture, the authors of the article use the general concept of the theorem, since an exact reproduction of the formula does not give the desired results.

Therefore, we will focus on the main idea of the theorem: a function that depends on many variables can be represented as a composition of continuous functions that depend on one variable.

Any neural network can be represented as a function that depends on many parameters (in our case, we have 16 parameters). Therefore, according to the theorem, we can divide it into at least 16 functions with one parameter (univariate function).

At this point, some questions might arise because MLP also depends on activation functions that take one parameter. The number of activation functions is equal to the number of neurons, and the product of data with synaptic weights coming to the neuron is the parameter. So what is the difference?

KAN proposes to replace the weights with learnable functions. Thus, the activation functions are shifted from neurons to edges, and neurons will only act as adders (Fig. 4).

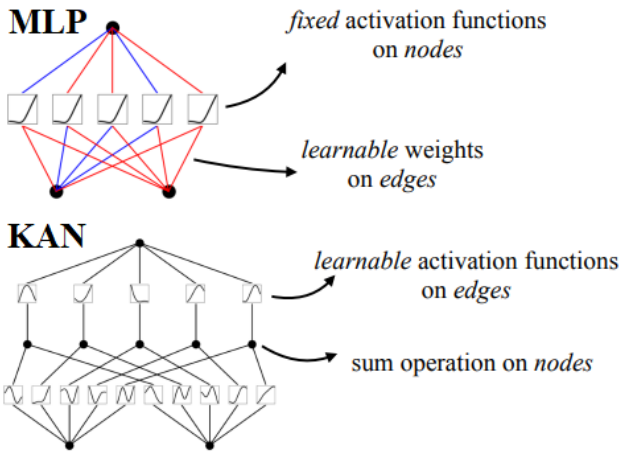

Figure 4. Visualisation of MLP and KAN models [15]

The difference is visually noticeable. Even with such a small model with two input parameters and one hidden layer, KAN has three times more activation functions. This feature confirms the need for KAN models to be much smaller than MLPs, as otherwise training will take a lot of time and resources.

Let's take a closer look at learnable features. Since all functions on the edges are univariate, we can parameterise them as B-spline curves [16].

A B-spline (or basis spline) is a function segmented by low-degree polynomials defined between given points. Let us consider the B-spline in (Fig. 5).

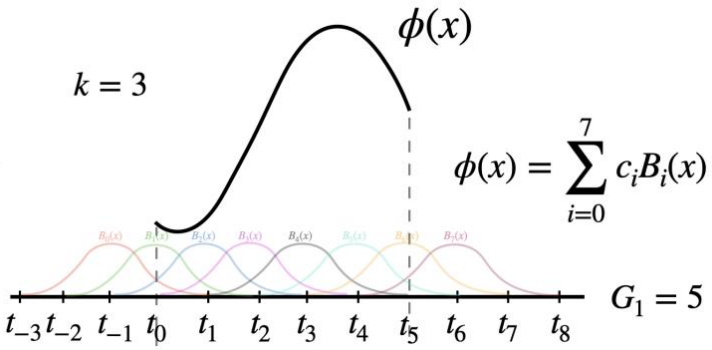

Figure 5. B-spline function on the edge [15]

In the figure, we see a continuous function on some interval defined by the nodes t_i. The value of G_1 is equal to the number of segments that make up the function. The parameter k is responsible for the degree of the polynomial that describes each segment of the curve. Thus, if $k = 3$, $B_i(x)$ is a third-degree polynomial. The training coefficient c_i affects the position and shape of each spline segment. Therefore, by changing the value of these coefficients, we can influence the overall shape of the function. Thus, c_i is analogous to the weighting coefficients in MLP.

Why B-Spline? The main advantage of B-Spline functions is local control. That is, a change in one coefficient affects the shape of only one segment of the spline. This feature solves the problem of "forgetting" in MLP (Fig. 6).

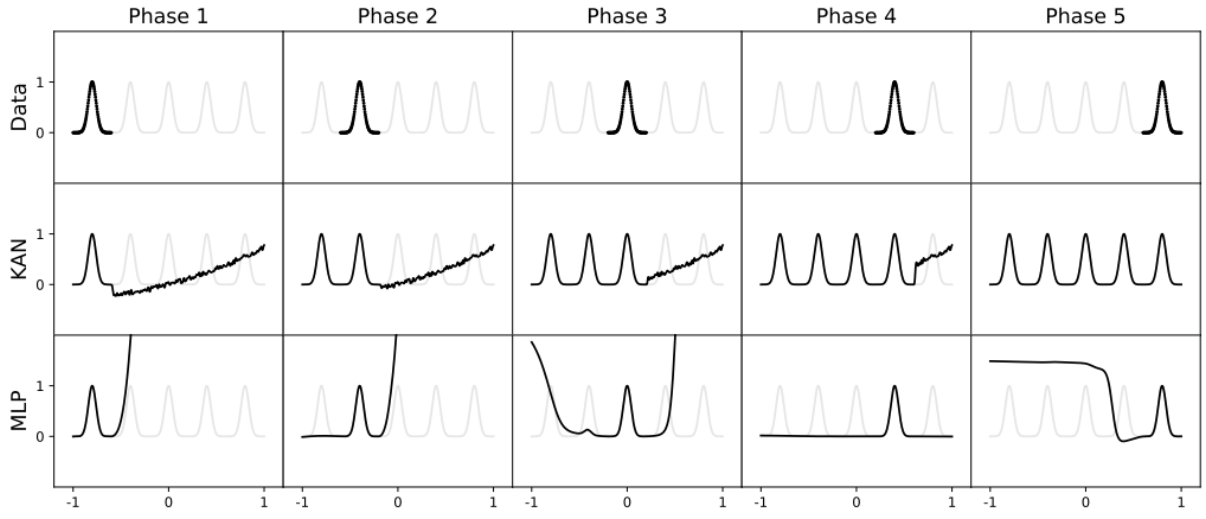

Figure 6. The effectiveness of MLP and KAN architectures additional training [15]

According to the data in the figure, we can conclude that additional MLP training is not effective. Because at each iteration of training with new data, MLP changes the weighting coefficients that affect the performance of the function over the entire range of values. This indicates the behaviour of global control, which causes problems in additional training of the model.

Each iteration in KAN has much less impact on the results of previous training. This is due to the large number of activation functions compared to MLP and the local control of each of them, which solves the problem of "forgetting" in the process of additional training.

Unlike MLP, KAN can increase the level of accuracy and local control using Grid Extension. The Grid Extension process involves increasing the number of segments into which the spline function is divided (Fig. 7).

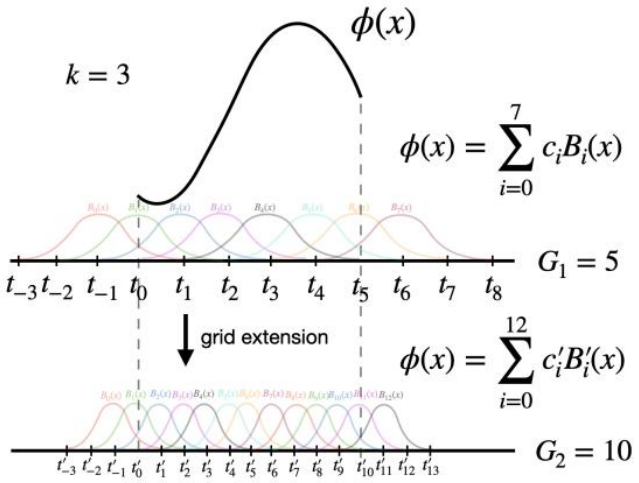

Figure 7. Grid extension y KAN [15]

Thus, the result of one function is influenced by a total of 12 basis functions. We determine the degree of discreteness of the spline function in Grid Extension ourselves. When choosing a value, one should keep in mind that with an increase in the number of segments, the training time and resource requirements will increase, since this operation significantly raises the number of calculations.

Pruning and Symbolisation. To reduce the time of the neural network operation, different methods of model optimisation are used. The main method in MLP is Pruning. The purpose of Pruning is to reduce the number of connections in the model. The fewer connections and neurons there are in the neural network, the fewer stages of computation the input data will go through.

During pruning in MLP, we search for neurons or connections that make no or very little contribution to the final result of the neural network. The marked elements are removed and they reduce the size of the model.

In addition to the above-described Pruning, KAN offers another approach that is called Symbolification. During training, some functions on the edges can take on the same or very close to the standard functions shape. Despite the similarity in shape, these functions are still splines. Obviously, calculating the conditional sin(x) is much simpler than the sum of the basis functions with weighting coefficients. Therefore, Symbolification suggests replacing splines with simpler functions: x2, x3, sin(x), cos(x), tan(x), ln(x), etc.

It is worth noting that it is desirable to use Symbolification on trained models. This is because replacing learnable functions with static ones will significantly reduce the efficiency of further training.

5. PROCEDURE FOR ANALYSING MODELS

The models will be analysed according to the following parameters: training time, accuracy metrics, and the amount of training data. The accuracy metrics include: accuracy_score, f1_score, roc_auc_score. Accuracy_score determines the degree of matching between the set returned by the model and the set of true values. F1_score determines the average value of the accuracy and recall of the model. Roc_auc_score summarises the classifier's performance across all possible classification thresholds.

The dataset on which the model will be trained and tested contains a total of 180 thousand rows. We will allocate 20% of each sample for testing. To evaluate the models more objectively, we will test them on a different number of samples: 10%, 30%, 50%, 70%, 100%.

6. TRAINING AND TESTING OF MODELS

All calculations will be performed on a PC with the following characteristics:

1. RAM: 16gb Ballistix 2666mhz.
2. CPU: Intel Core i5-10400f 2.9Ghz
3. HDD

The MLP architecture will be implemented by the MLPClassifier module of the sklearn library [19]. We set the following hyperparameters for the model:

1. activation: 'logistic'
2. hidden_layer_sizes: 100
3. batch_size: auto, 32 for a sample of 20% of the total data
4. max_iter: 200
5. solver: adam

To train the KAN neural network, we will use the pykan library [18] by the authors of the original article. The library was developed based on the basic classes of the Pytorch framework. For initialisation, the following parameters were used:

1. width: [16, 3, 1]. The first and last elements of the array determine the dimensionality of the input and output data, respectively. The indices of the elements inside determine the order of the layers, and their values determine the number of neurons in a layer.
2. grid: 3. The grid dimension.
3. k: 3. The degree of the polynomial.
4. noise_scale: 0.1. Estimation of the data noise level.

When training, we set the following parameters:

1. opt: 'LBFGS'
2. steps: 50
3. batch: number of rows in the test sample
4. loss_fn: BCEWithLogitLoss
5. lr: 0.1

The results of both architectures are shown in the table.

Table 1

Results of the architectures.

Architecture	Data amount	Training time in seconds	Accuracy	F1	Roc Auc
1	2	3	4	5	6
MLP	10%	31.2	73.6	73.7	73.6
KAN	10%	36.8	76.7	75.9	76.8
MLP	30%	71.1	76.6	76.6	76.6

The end of the table

1	2	3	4	5	6
KAN	30%	101.6	78.4	78.8	78.4
MLP	50%	113.5	77.8	77.1	77.8
KAN	50%	172.9	77.5	76.2	77.4
MLP	70%	153.7	78.7	78.2	78.6
KAN	70%	251.1	77.6	76.1	77.5
MLP	100%	222.9	78.2	78.0	78.2
KAN	100%	377.5	77.4	76.4	77.4

7. CONCLUSIONS

We solved the problem of binary classification using two neural network architectures on different amounts of data. Having analysed the results of both networks, we can conclude that MLP is faster in training and outperforms KAN by 0.5–1.5% on larger data sets: 50%, 70%, 100%. KAN shows more accurate results while training on 10% and 30% samples, thus outperforming MLP by 2–3%. Taking into account the novelty of KAN and the future optimisation of existing libraries and development of new ones, this architecture can replace MLP for tasks with a small amount of training data. Considering the small size of the models and optimisation algorithms, KAN can run more efficiently on devices with low computing power, making them more accessible. In the future, KAN layers can be used in architectures with fully connected layers of other networks ranging from CNNs to LLMs.

References

1. Evan D., Nick N., Josh W. (2024). The Regents of the University of Michigan. Mind control prothesis. Available at: https://spotlight.engin.umich.edu/mind-control-prosthesis/ (accessed 05.10.2024).
2. Nishal P., Matthew S., Nick H., Foram K., (2024). A flexible intracortical brain-computer interface for typing using finger movements. Cord Spring Harbor laboratory. https://doi.org/10.1101/2024.04.22.590630
3. Musk E., Neuralink (2019) An Integrated Brain-Machine Interface Platform With Thousands of Channels. Journal of medical Internet research, vol. 21, no. 10. Available at: https://pmc.ncbi.nlm.nih.gov/articles/PMC6914248/ (accessed: 06.10.2024). https://doi.org/10.2196/16194
4. Willett F. R., Avansino D. T., Hochberg L. R. et al. (2021) High-performance brain-to-text communication via handwriting. Nature 593, 249–254. https://doi.org/10.1038/s41586-021-03506-2
5. Pastukh O., Yatsyshyn V., (2023) Brain-Computer Interaction Neurointerface Based On Artificial Intelligence And Its Parallel Programming Using High-Performance Calculation On Cluster Mobile Devices. Scientific Journal of the Ternopil National Technical University, no. 4. https://doi.org/10.33108/visnyk_tntu2023.04.026
6. Levi T, Bonifazi P, Massobrio P and Chiappalone M., (2018) Editorial: Closed-Loop Systems for Next-Generation Neuroprostheses. Front. Neurosci. 12:26. https://doi.org/10.3389/fnins.2018.00026
7. Agrawal A., Ray B., & Bal R. Redolfi Riva E., Micera S. (2021). Progress and challenges of implantable neural interfaces based on nature-derived materials. 2018. Bioelectron Med 7, 6. https://doi.org/10.1186/s42234-021-00067-7
8. Marco V., Leigh R. Applications of neural interfaces in prosthetic control. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 26 (1), 101–110. Available at: https://doi.org/10.1109/TNSRE.2018.2792051 (accessed: 08.10.1024).
9. Joon Y. L, Ssang H. S. (2024) EEG-Based Emotion Recognition Using Deep Learning Model for Workers Safety. Nanotechnology Perceptions, vol. 20, no. S2. https://doi.org/10.62441/nano-ntp.v20iS2.25
10. Apicella A., Arpaia P., Giugliano S., Mastrati G., Moccaldi N. (2022) High-wearable EEG-based transducer for engagement detection in pediatric rehabilitation Brain-Computer Interfaces, 9 (3), pp. 129–139. https://doi.org/10.1080/2326263X.2021.2015149
11. Memmott T., Kocanaogullari A., Lawhead M., Fried-Oken M., Oken B. (2021) BciPy: brain-computer interface software in Python. Brain-Computer Interfaces, 8 (4), pp. 137–153. https://doi.org/10.1080/2326263X.2021.1878727
12. McFarland D. J., Norman S. L., Sarnacki W. A., Reinkensmeyer D. J., Wolpaw J. R. (2020) BCI-based sensorimotor rhythm training can affect individuated finger movements. Brain-Computer Interfaces 7 (1–2), pp. 38–46. https://doi.org/10.1080/2326263X.2020.1763060

13. Chen C., Chen P., Belkacem A. N., Wang C., Ming D. 2020. Neural activities classification of left and right finger gestures during motor execution and motor imagery. Brain-Computer Interfaces pp. 1–11. https://doi.org/10.1080/2326263X.2020.1782124
14. Cross-Modulated Amplitudes and Frequencies of Epileptic EEG. Available at: https://www.researchgate.net/publication/335581366_Cross Modulated_Amplitudes_and_Frequencies_of_Epileptic_EEG (accessed: 0.7.10.2024).
15. Ziming L., Yixuan W., Sachin V., Fabian R., James H., Marin S., Thomas Y. H., Max T. 2024. KAN: Kolmogorov–Arnold Networks. Available at: doi.org/10.48550/arXiv.2404.19756 (accessed: 0.7.10.2024 - 25.10.2024).
16. B-spline curve. Available at: https://web.mit.edu/hyperbook/Patrikalakis-Maekawa-Cho/node17.html (accessed: 10.10.2024).
17. Feed forward network diagram library. Available at: https://github.com/martisak/dotnets (accessed: 22.10.2024).
18. Pykan library documentation. Available at: https://kindxiaoming.github.io/pykan/modules.html (accessed: 20.10.2024 - 25.10.2024).
19. Sklearn python machine learning library. MLPClassifier documentation. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html (accessed: 21.10.2024)

УДК 681.3

ПОРІВНЯЛЬНИЙ АНАЛІЗ НЕЙРОМЕРЕЖЕВИХ АРХІТЕКТУР MLP ТА KAN У НЕЙРОІНТЕРФЕЙСНИХ ТЕХНОЛОГІЯХ

Юрій Петров; Олег Пастух

Тернопільський національний технічний університет імені Івана Пулюя, Тернопіль, Україна

Резюме. *Досліджено актуальну проблему нейроінтерфейсних технологій, особливо в контексті протезування людей з обмеженими можливостями. Зокрема, проаналізовано можливості нейроінтерфейсів у відновленні функцій кінцівок, що дозволяє користувачам протезів легше виконувати повсякденні задачі. Було розглянуто дві нейромережеві архітектури – MLP (multilayer perceptron) та KAN (Kolmogorov Arnold network), які застосувалися для класифікації рухів пальців на основі мозкових сигналів. Для формування датасету використано 16-канальний електроенцефалограф, за допомогою якого реєструвалися зміни електричної активності мозку під час виконання учасником послідовних рухів згинання та розгинання пальців протягом 2 хвилин. Такі дії виконувались окремо для великого та вказівного пальців, що дозволило розрізняти сигнали різних рухових активностей. Схема розташування електродів охоплювала відповідні ділянки мозку, пов'язані з руховою активністю кожного з пальців. Отримано результати ефективності роботи та навчання обох архітектур на різних вибірках даних від 10% до 100% від загального датасету в 180 тисяч рядків. Загалом аналіз показав невелику перевагу KAN 2–3% у точності на малих обсягах тренувальних даних (10%, 30%), що пов'язано з кращою здатністю до узагальнення за обмежених даних. Також сильною стороною є значно менший розмір моделей KAN завдяки використанню B-сплайн функцій та можливостям символізації, але тривалий час та вищі потреби до обчислювальних ресурсів під час навчання через більшу кількість функцій активації. MLP перемагає у швидкості тренування та показує точніші результати на 0.5%–1.5% при великих обсягах даних (50%, 70%, 100%), що робить його ефективнішим для роботи з великими датасетами. Звідси випливають перспективи розвитку нової архітектури у сфері біонічних протезів та інших технологій на основі нейроінтерфейсів, що сприятиме покращенню якості життя людей з фізичними обмеженнями. Враховуючи новизну KAN та майбутню оптимізацію існуючих бібліотек, ця архітектура може замінити MLP для задач з невеликою кількістю тренувальних даних.*

Ключові слова: *нейромережі, нейроінтерфейс, MLP, KAN, людино-машинна взаємодія, штучний інтелект, машинне навчання.*

https://doi.org/10.33108/visnyk_tntu2025.03.107 *Отримано 07.08.2025*