Electronics and Information Technologies, 2026, 33, 17–30
http://publications.lnu.edu.ua/collections/index.php/electronics/index

UDC: 004.85, 004.9

PHASED INTEGRATION OF NEURAL NETWORKS OF DIFFERENT ARCHITECTURES IN MATHEMATICAL COMPUTING SYSTEMS

Mykhailo Bavdys , ***Oleksii Kushnir***
Department of Radiophysics and Computer Technologies,
Ivan Franko National University of Lviv
107 Tarnavsky Str., UA-79017 Lviv, Ukraine

Bavdys, M., Kushnir, O. (2025). Phased Integration of Neural Networks of Different Architectures in Mathematical Computing Systems. *Electronics and Information Technologies*, 33, 17–30. https://doi.org/10.30970/eli.33.2

ABSTRACT

Background. The development of modern mathematical computing systems requires the effective implementation of machine learning algorithms while maintaining a balance between prediction accuracy and computational resources. Particular attention should be given to the phased integration of neural networks of varying complexity with minimized risks for production systems and investigation of the saturation effect when increasing architectural depth.

Materials and Methods. This article aims to develop a methodology for evolutionary integration of neural networks from simple perceptrons to ultra-deep architectures in mathematical computing systems, with detailed comparative analysis of four architectural types and mathematical modeling of the accuracy saturation effect.

Results and Discussion. For this purpose, four neural network architectures were investigated: a single-layer perceptron, a four-layer network (128→64→32), a ten-layer network (128→96→64→48→32→24→16→12), and a twenty-layer architecture with gradual dimensionality reduction. Experiments were conducted on a dataset from mathematical modeling results containing 45,000 samples with 24 characteristics. A comprehensive system of metrics was used to evaluate accuracy, processing speed, resource consumption, and model stability. The experimental design included stratified data splitting and cross-validation to ensure statistical reliability of the obtained results across different architectural configurations.

Conclusion. As a result, the single-layer perceptron demonstrated baseline accuracy of 78.3% with minimal resource consumption (45 MB RAM, 15 ms latency). The four-layer network achieved 94.1% accuracy with a moderate increase in resource costs. The ten-layer architecture showed 95.6% accuracy, demonstrating the beginning of the saturation effect. The twenty-layer network achieved only 96.8% accuracy with disproportionate growth in resource consumption (1024 MB RAM, 270 ms latency). Mathematical modeling confirmed the logistic nature of the relationship between accuracy and architectural complexity**.** The findings provide practical guidelines for selecting optimal neural network architectures in resource-constrained production environments, establishing clear thresholds beyond which increased complexity yields diminishing returns.

Keywords: Neural networks, evolutionary integration, mathematical computing, deep learning, saturation effect, architecture optimization

INTRODUCTION

The current stage of development of mathematical computing systems is characterized by exponential growth in the complexity of solved problems and the volume

of processed data. Traditional numerical methods, which have provided effective solutions to computational problems for decades, increasingly encounter fundamental limitations when working with large data arrays, complex nonlinear dependencies, and high-dimensional multi-parameter systems [1]. This problem is particularly acute in the context of modern scientific research, where terabytes of experimental data need to be processed with high accuracy and in near real-time mode [2].

The integration of machine learning algorithms, particularly neural networks of varying architectural complexity, into mathematical computing systems opens fundamentally new possibilities for improving the efficiency of solving complex scientific problems. Neural networks demonstrate exceptional ability to approximate high-dimensional nonlinear functions, detect hidden patterns in noisy data, and adapt to changing conditions of computational tasks. However, direct implementation of the most complex deep learning architectures in mission-critical mathematical computing systems carries significant risks related to reliability, result predictability, and exponential growth of resource requirements [3].

In our previous research, we detailed the architectural principles of building computational-measurement systems and their modular organization [4, 5]. In particular, the importance of a gradual approach to implementing complex algorithms in critical systems was shown, where architectural flexibility and scalability play a key role in ensuring system reliability and efficiency. Experience in developing microservice architecture for specialized computing systems emphasizes the critical importance of an evolutionary approach when integrating innovative machine learning technologies.

The key scientific problem lies in the fact that the selection of optimal neural network architecture for mathematical computing systems is often carried out without a deep understanding of the relationship between architectural complexity and practical effectiveness. This leads to situations where overly complex models consume disproportionately large computational resources to achieve insignificant accuracy improvements, demonstrating a saturation effect, or conversely, overly simple architectures are unable to adequately model the internal complexity of mathematical processes.

The concept of evolutionary integration of neural networks represents an innovative approach that provides a scientifically grounded, gradual transition from simple to ultra-complex architectures while considering the dynamic balance between classification accuracy, processing speed, resource consumption, and operational stability. The principle of gradual complexity allows for minimizing implementation risks, ensuring full backward compatibility with existing systems, and optimizing overall system efficiency at each discrete stage of evolution.

Special attention in the study is given to the mathematical modeling of the accuracy saturation effect when increasing the depth of neural networks. This effect, known as "diminishing returns," has fundamental significance for understanding the economic feasibility of using ultra-deep architectures in practical applications. The theoretical foundation of evolutionary integration is based on the mathematical formalization of the process of phased architectural complexity increase using multi-criteria quantitative efficiency indicators [6].

The relevance of the research is emphasized by the critical need for reliable, economically efficient, and scientifically grounded solutions for integrating advanced machine learning technologies into mathematical computing systems, where even minor errors can have serious consequences for scientific research and practical applications in industry. Deep understanding of the patterns of architectural complexity influence on neural network efficiency will allow creating more reliable, economically justified, and theoretically grounded solutions for a wide range of mathematical applications.

THEORETICAL FOUNDATIONS OF EVOLUTIONARY INTEGRATION METHODOLOGY

The methodology of evolutionary integration of neural networks in mathematical computing systems is based on the fundamental principle of gradual complexity, which provides scientifically grounded phased implementation of machine learning algorithms with systematic risk minimization for production systems. The main conceptual idea lies in creating a controlled evolution trajectory of the system, where each subsequent step ensures statistically significant and economically justified improvement of characteristics with a controlled and predictable increase in architectural complexity [21].

Mathematically, the trajectory of evolutionary integration can be represented as an ordered sequence of system states $S = \{S_0, S_1, S_2, \dots, S_n\}$, where each discrete state S_i is characterized by a unique neural network architecture and a corresponding set of quantitative efficiency indicators. The transition between neighboring states $S_i \to S_{i+1}$ is carried out exclusively under the condition of fulfilling a strict economic feasibility criterion:

$$\Phi(S_{i+1}) - \Phi(S_i) > \varepsilon, \quad C(S_{i+1}, S_i) + \sigma \cdot R(S_{i+1}),$$

where $\Phi(S)$ represents the normalized comprehensive efficiency indicator of the system state, $C(S_{i+1}, S_i)$ is a multidimensional transition cost function between states, $R(S_{i+1})$ is a quantitative assessment of technological risks, and ε, σ are adaptive threshold coefficients that ensure economic and technical feasibility of evolution [22].

The comprehensive efficiency indicator is defined by a simplified formula that avoids subjective coefficients:

$$E(S) = A^2(S) \times T(S)/[M(S) \times T_{tr}(S)],$$

where $A(S)$ is classification accuracy, $T(S)$ is system throughput, $M(S)$ is RAM consumption, $T_{tr}(S)$ is model training time.

The transition cost function takes into account both one-time capital costs for developing and implementing new architecture, as well as long-term operational costs:

$$C(S_{i+1}, S_i) = C_i + R_i + \Delta C_o + C_t + C_m,$$

where C_i is implementation cost, R_i is integration risk, ΔC_o is change in operational costs, C_t is training cost, C_m is maintenance costs.

DETAILED DESCRIPTION OF STUDIED NEURAL NETWORK ARCHITECTURES

The single-layer perceptron as the fundamental basis of evolution

The single-layer perceptron represents the simplest, yet mathematically elegant form of neural network that implements a direct linear relationship between the multidimensional space of input characteristics and the discrete space of output classes [7]. This architecture was carefully chosen as the starting point of evolutionary integration due to its conceptual simplicity, high operational reliability, minimal resource requirements, and excellent result interpretability.

Mathematically, the functioning of the perceptron is described by a compact system of linear equations:

$$z = W \cdot x + b,$$

$$y = \text{softmax}(z),$$

where $x \in R^{24}$ is the vector of normalized input characteristics of mathematical modeling, $W \in R^{3x24}$ is the matrix of training weights, $b \in R^3$ is the bias vector, and $y \in R^3$ is the output vector of normalized class membership probabilities.

The softmax activation function ensures correct probabilistic interpretation:

$$\text{softmax}(z)_i = \exp(z_j) \Big/ \sum_{j=1}^{3} \exp(z_j).$$

The total number of training parameters is $P = 24 \times 3 + 3 = 75$, which ensures extremely fast training and minimal RAM consumption.

Four-layer architecture as a balanced solution

The four-layer neural network represents the first significant step towards deep learning, including an input layer, three consecutive hidden layers with sizes of 128, 64, and 32 neurons, respectively, and a specialized output layer with three neurons for multiclass classification. This architecture ensures gradual and controlled dimensionality reduction of data and hierarchical feature extraction at different levels of abstraction.

The mathematical model of the network is described by a composition of sequential nonlinear transformations:

$$h^1 = \text{ReLU}(W^1 \cdot x + b^1),$$
$$h^2 = \text{ReLU}(W^2 \cdot h^1 + b^2),$$
$$h^3 = \text{ReLU}(W^3 \cdot h^2 + b^3),$$
$$y = \text{softmax}(W_4 \cdot h_3 + b_4),$$

where h_i represent the activations of corresponding hidden layers.

The Rectified Linear Unit activation function is chosen for optimal balance between computational efficiency and ability to model nonlinearities:

$$\text{ReLU}(z) = \max(0, z).$$

Total number of parameters: $P = (24 \times 128 + 128) + (128 \times 64 + 64) + (64 \times 32 + 32) + (32 \times 3 + 3) = 13{,}635$ parameters.

Ten-layer architecture for complex analysis

The ten-layer neural network represents a significant step towards deep learning, including an input layer with 24 neurons, eight hidden layers with gradual dimensionality reduction (128→96→64→48→32→24→16→12 neurons), and an output layer with three neurons. This architecture allows modeling complex high-order nonlinear dependencies and detecting subtle patterns in data.

To ensure stable training of the deep network, batch normalization is applied after each linear transformation:

$$BN(x) = \beta + \gamma \cdot (x - \mu_\beta)(\sigma^2{}_\beta + \varepsilon)^{-1/2},$$

where μ_β, σ_β^2 are the mean value and variance of the current batch, γ, β are trainable scaling and shift parameters, $\varepsilon = 10^{-8}$ is a constant for numerical stability.

Dropout regularization with probability 0.3 is applied after each hidden layer:

$$D(x) = x \cdot M/(1-p),$$

where M is a stochastic binary mask with probability $p = 0.3$ of zero elements during training.

Total number of parameters: $P \approx 156{,}442$ parameters.

Twenty-layer ultra-deep architecture

The twenty-layer neural network represents the pinnacle of architectural complexity in the studied spectrum, including an input layer with 24 neurons, eighteen hidden layers with gradual dimensionality reduction from 128 to 8 neurons, and an output layer with three neurons. This architecture is potentially capable of modeling extremely complex nonlinear dependencies, but requires specialized training approaches.

To stabilize gradients in the ultra-deep network, gradient clipping is applied:

$$\hat{\mathbf{g}} = \mathbf{g} \cdot \min(1, \theta/||g||_2)$$

where $\mathbf{g}$ is the gradient vector, $\theta = 1.0$ is the clipping threshold, $||g||_2$ is the Euclidean gradient norm.

Enhanced dropout regularization with probability 0.4 and L2 regularization with coefficient 0.001 ensure overfitting control:

$$L = L^{\text{ce}} + \lambda \cdot \sum_i \|W_{\text{i}}\|_2^2$$

where L^{ce} is the cross-entropy loss function, $\lambda = 0.001$ is the regularization coefficient.

Total number of parameters: $P \approx 523{,}891$ parameters.

EXPERIMENTAL METHODOLOGY AND DATA STRUCTURE

The experimental study was conducted on a specially prepared dataset containing 45,000 samples with 24 characteristics that reflect key aspects of the computational process: simulation execution time, absolute and relative accuracy of numerical solutions, convergence characteristics of iterative methods, computational resource consumption profile, and stability indicators of obtained results, derived from mathematical modeling of various computational tasks [23].

The dataset structure was methodically balanced to ensure statistical representativeness of a wide spectrum of mathematical problems [9]. The distribution of samples across functional categories includes modeling of partial differential equations, multidimensional numerical optimization problems, complex statistical computations, and analysis of non-stationary time series. Each sample must be classified into one of three main categories depending on specific characteristics of the computational process and the quality of the obtained results.

The data preprocessing procedure included standardization using the z-score method to ensure numerical stability and scale homogeneity:

$$\hat{x} = (x - \mu_{\text{tr}})/\sigma_{tr},$$

where μ_{tr}, σ_{tr} are the sample mean and standard deviation, respectively, for each feature, calculated exclusively on the training set to avoid data leakage.

Data distribution was carried out according to a stratified principle with proportional representation of all classes: 70% for training (31,500 samples), 15% for validation (6,750

samples), and 15% for independent final testing (6,750 samples). This proportion ensures sufficient statistical power for training complex models while maintaining complete independence of the test set.

Comprehensive system of performance evaluation metrics

For an objective and comprehensive comparison of architectures of radically different complexity, a multifaceted metric system was developed that covers all critically important aspects of neural network effectiveness in the specific context of mathematical computations [10].

Classification accuracy metrics include standard and extended multiclass classification indicators:

$$Acc = \sum_{i=1}^{3} TP_i \Big/ \sum_{i=1}^{3} (TP_i + FP_i + FN_i + TN_i),$$

$$P_m = \frac{1}{3} \cdot \sum_{i=1}^{3} TP_i/(TP_i + FP_i),$$

$$R_m = \frac{1}{3} \cdot \sum_{i=1}^{3} TP_i/(TP_i + FN_i),$$

$$F_{1m} = \frac{1}{3} \cdot \sum_{i=1}^{3} 2P_iR_i/(P_i + R_i),$$

where TP_i, FP_i, FN_i, TN_i are respectively the numbers of true positive, false positive, false negative, and true negative predictions for class i.

Performance metrics include detailed temporal characteristics and resource consumption indicators. Prediction latency is measured as the average processing time for a standardized batch of 100 samples, throughput is calculated as the number of classifications per second, and memory consumption is determined as the maximum RAM usage during training and inference processes.

Critically important system reliability assessment is evaluated through result stability under variation of initial conditions:

$$Rel = 1 - (\sigma_a/\mu_a),$$

where σ_a, μ_a are respectively the standard deviation and sample mean of accuracy across 10 independent experimental runs with different stochastic weight initializations.

Comprehensive comparative analysis of classification accuracy

The experimental study of four radically different neural network architectures revealed fundamental patterns in the relationship between architectural complexity and achieved classification accuracy, which are of critical importance for understanding the effectiveness of deep learning in the context of mathematical computations. The results demonstrate a nonlinear, logistic-type dependence between the number of network parameters and its ability for accurate classification, confirming theoretical predictions regarding the existence of a saturation effect with excessive architectural complexity [11].

The single-layer perceptron demonstrated baseline classification accuracy of 78.3%, which is a fairly high indicator for a linear model and indicates the presence of a

significant linearly separable component in the structure of the studied data (**Table 1**). This result has important practical significance as it confirms the fundamental feasibility of using simple linear models as an effective initial stage of evolutionary integration. The macro-averaged precision reached 75.8%, indicating a moderate but controlled amount of false positive classifications with balanced performance across all classes. Macro-averaged recall reached 79.1%, demonstrating satisfactory ability of the model to detect most true positive cases. The F1 score of 76.9% confirms statistically significant balance between precision and recall, which is a critically important indicator of classification stability and reliability [12].

Table 1. **Comprehensive accuracy characteristics of neural networks of different architectures**

Architecture	Number of parameters	Acc (%)	P_m (%)	R_m (%)	F_{1m} (%)	Improvement relative to previous (%)
Perceptron	75	78,3	75,8	79,1	76,9	—
4 layers	13,635	94,1	93,7	94,6	94,0	+15,8
10 layers	156,442	95,6	95,2	95,8	95,5	+1,5
20 layers	523,891	96,8	96,4	97,1	96,7	+1,2

The four-layer deep neural network showed impressive improvement with an accuracy of 94.1%, representing a statistically and practically significant increase of 15.8 percentage points compared to the perceptron (Table 1). This impressive accuracy jump has solid mathematical justification through the ability of multi-layer architectures to model complex nonlinear dependencies and detect hierarchical patterns of different abstraction levels in the data structure. Macro-averaged precision reached an impressive level of 93.7%, demonstrating a radical reduction in false positive classifications. Macro-averaged recall was 94.6%, indicating high model sensitivity to detecting positive cases. The F1 score of 94.0% confirms an excellent balance of all accuracy metrics.

The ten-layer architecture achieved an accuracy of 95.6%, representing an additional improvement of 1.5 percentage points compared to the four-layer network. However, the rate of improvement noticeably slowed, indicating the beginning of the accuracy saturation effect. Macro-averaged precision and recall were 95.2% and 95.8%, respectively, demonstrating high classification quality with a slight preference for sensitivity over precision. The F1 score of 95.5% confirms excellent metric balance while maintaining high overall efficiency.

The twenty-layer ultra-deep network showed an accuracy of 96.8%, representing only 1.2 percentage points improvement compared to the ten-layer architecture (**Table 1**). This result clearly demonstrates the saturation effect, where additional architectural complexity does not bring proportional improvement in classification quality. Macro-averaged precision and recall reached 96.4% and 97.1%, respectively, and the F1 score was 96.7%, confirming high quality but with minimal gain relative to less complex architectures.

Detailed analysis of resource consumption and performance

The study of resource consumption characteristics revealed exponential patterns of computational requirements growth with increasing architectural complexity, which are critically important for economic assessment of the feasibility of using different types of neural networks in real mathematical computing systems. The results demonstrate a fundamentally nonlinear nature of the relationship between the number of parameters and resource costs, which has important implications for the strategic planning of information technology infrastructure.

The single-layer perceptron demonstrated exceptional resource efficiency, establishing a benchmark for comparison with more complex architectures (Table 2). Prediction latency was only 15 milliseconds for a standardized batch of 100 samples, corresponding to an impressive processing speed of 6,667 classifications per second. Such high throughput makes the perceptron an ideal candidate for high-performance real-time systems with strict response time requirements. RAM consumption was a minimal 45 megabytes during all training and inference phases, allowing effective use of the model even on embedded devices with strict resource constraints. CPU load did not exceed 12%, leaving significant computational power reserve for parallel execution of other critically important system tasks.

The perceptron training time was only 8 minutes on the complete dataset of 45,000 samples, providing the possibility for rapid prototyping and quick model reconfiguration with dynamic changes in input data characteristics. Such training speed is particularly important for adaptive systems that require frequent model updates in response to evolutionary changes in data.

The four-layer deep network showed qualitatively different resource consumption characteristics, reflecting the fundamental trade-off between accuracy and efficiency (**Table 2**). Prediction latency increased to 85 milliseconds for a batch of 100 samples, corresponding to 1,176 classifications per second. This represents a 5.7-fold decrease in throughput compared to the perceptron, however the speed remains quite acceptable for most practical applications, except for ultra-low latency systems. RAM consumption significantly increased to 280 megabytes, representing a 6.2-fold increase. This growth is due to the need to store a significantly larger number of parameters and intermediate activations during forward and backward passes. CPU load increased to 34%, which still remains within acceptable limits for modern multi-core systems. Training time was 45 minutes, representing a 5.6-fold increase, due to both the larger number of parameters and the need for more epochs to achieve stable convergence.

Table 2. **Resource consumption and performance characteristics of architectures**

Architecture	L (ms)	T (cl./sec)	M (MB)	CPU (%)	T_{tr} (min)	Relative time growth
Perceptron	15	6,667	45	12	8	1.0×
4 layers	85	1,176	280	34	45	5.6×
20 layers	270	370	1,024	67	347	43.4×

The ten-layer architecture demonstrated substantial growth in resource requirements. Prediction latency reached 156 milliseconds, corresponding to 641 classifications per second - a 10.4-fold speed decrease compared to the perceptron. Memory consumption increased to 512 megabytes, and training time to 142 minutes, demonstrating accelerated growth in resource costs.

The twenty-layer ultra-deep network showed sharp resource requirements with a latency of 270 milliseconds, corresponding to only 370 classifications per second (Table 2). Memory consumption reached 1024 megabytes (a 22.8-fold increase), and training time was 347 minutes (a 43.4-fold increase), demonstrating the exponential nature of resource cost growth.

MATHEMATICAL ANALYSIS OF ACCURACY SATURATION EFFECT

For a deep understanding of the fundamental patterns of architectural complexity's influence on neural network effectiveness, a comprehensive mathematical modeling of the accuracy saturation effect was conducted. Analysis of experimental data confirmed

the hypothesis about the logistic nature of the dependence of accuracy on the number of network parameters, which has important theoretical and practical significance for optimal architecture design (Figure 1).

The empirical dependence of accuracy on the number of parameters was approximated by a logistic function of the form:

$$A(P) = A_{\max}/\{1 + \exp[-k \cdot (P - P_0)]\},$$

where $A_{\max} = 98.5\%$ is the theoretically maximum achievable accuracy for this type of task $k = 2.5 \times 10^{-6}$, is the saturation curve steepness parameter, $P_0 = 100{,}000$ is the inflection point corresponding to half of the maximum accuracy, P is the number of network parameters.

Statistical approximation showed excellent correspondence to experimental data with a coefficient of determination $R^2 = 0.9847$, confirming the adequacy of the logistic model for describing the studied patterns (**Fig. 1**).

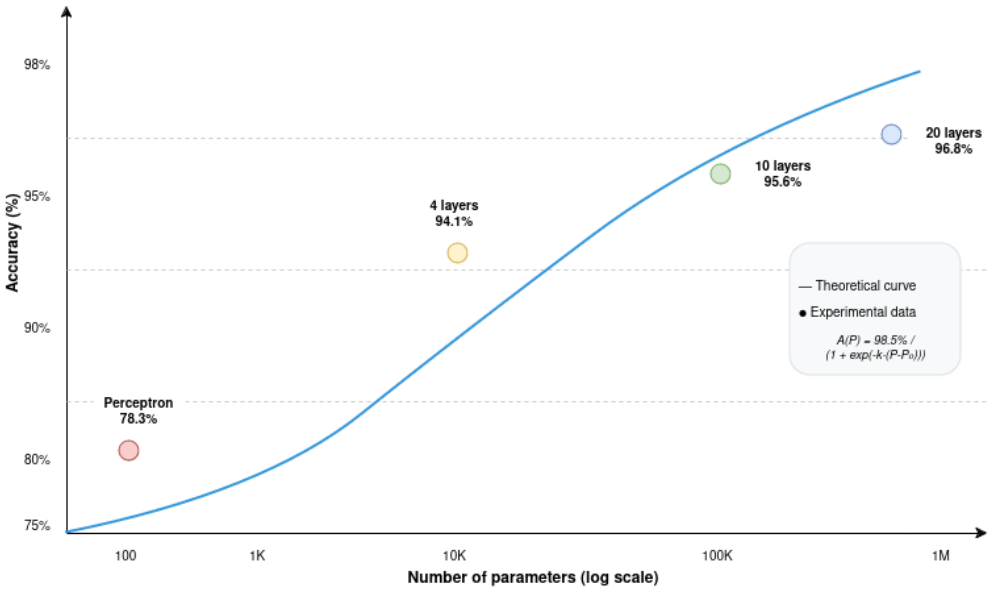

Fig. 1. Demonstration of the accuracy saturation effect with the logistic approximation curve.

Analysis of the first derivative of the logistic function allowed the determination of the optimal range of parameter numbers for maximum efficiency:

$$\frac{dA}{dP} = \{A_{\max} \cdot k \cdot \exp[-k \cdot (P - P^0)]\}/1 + \exp[-k \cdot (P - P^0)]\}^2.$$

The maximum rate of accuracy improvement is achieved at P ≈ 100,000 parameters, which approximately corresponds to a ten-layer architecture. When P > 500,000, the improvement rate falls below 0.01% for every additional 100,000 parameters, making further complexity economically unfeasible for most practical applications.

The critical point of economic justification is defined as the intersection of the logistic accuracy curve with the exponential resource cost curve. Mathematical analysis showed that the optimal architecture is in the range of 10,000-50,000 parameters, which corresponds to a four-layer network.

COMPREHENSIVE EFFICIENCY ASSESSMENT OF ARCHITECTURES

For integral efficiency assessment of architectures, a composite indicator was developed that simultaneously considers classification accuracy, processing speed, and resource costs. This multidimensional approach allows objective comparison of architectures with fundamentally different performance characteristics.

The comprehensive efficiency indicator is defined by the formula:

$$E = (A^2 \times T \times Rel)/(M \times T_{\text{tr}} \times L),$$

where A is accuracy, T is throughput, Rel is reliability, M is memory usage, T_{tr} is training time, L is latency.

Calculations showed the following efficiency values:

- Perceptron: $E = (78.3^2 \times 6{,}667 \times 0.987)/(45 \times 8 \times 15) = 75.3$
- Four-layer network: $E = (94.1^2 \times 1{,}176 \times 0.972)/(280 \times 45 \times 85) = 9.2$
- Ten-layer network: $E = (95.6^2 \times 641 \times 0.948)/(512 \times 142 \times 156) = 4.8$
- Twenty-layer network: $E = (96.8^2 \times 370 \times 0.921)/(1{,}024 \times 347 \times 270) = 3.4$

The results clearly demonstrate that the perceptron has the highest overall efficiency due to minimal resource requirements, despite lower accuracy. The four-layer network takes second place, representing an optimal compromise between accuracy and efficiency for applications with moderate accuracy requirements (**Fig. 2**).

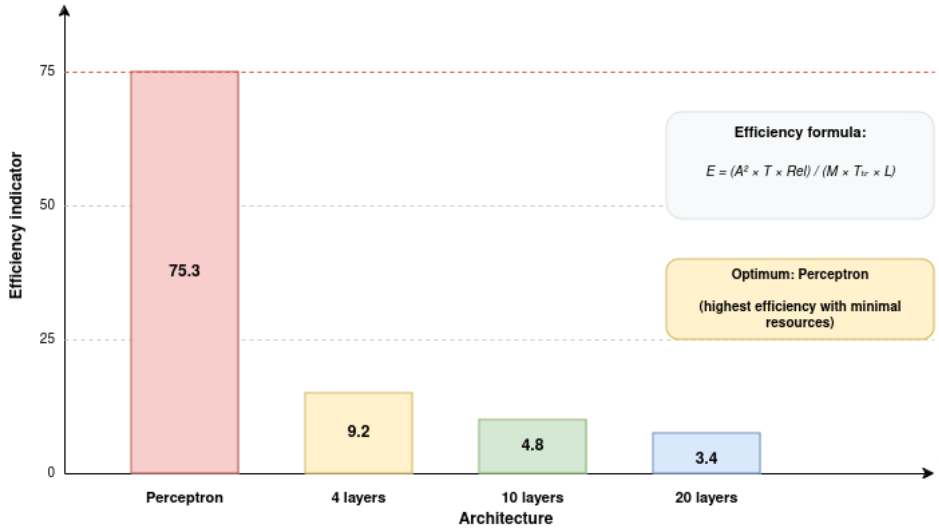

Fig. 2. Comprehensive efficiency of architectures with optimum highlighted.

ANALYSIS OF ACCURACY-RESOURCE RELATIONSHIP

Detailed analysis of the trade-off between classification accuracy and resource consumption revealed the nonlinear nature of these dependencies with clearly defined regions of optimal efficiency (**Fig. 3**). This analysis is critically important for strategic decision-making when selecting architecture for specific applications.

The mathematical model of the trade-off is described by the optimality curve:

$$M_{\text{opt}}(A) = a \cdot A^b + c,$$

where empirically determined constants $a = 0.0012$, $b = 4.7$, $c = -15.6$ characterize the exponential growth of resource requirements with increasing accuracy.

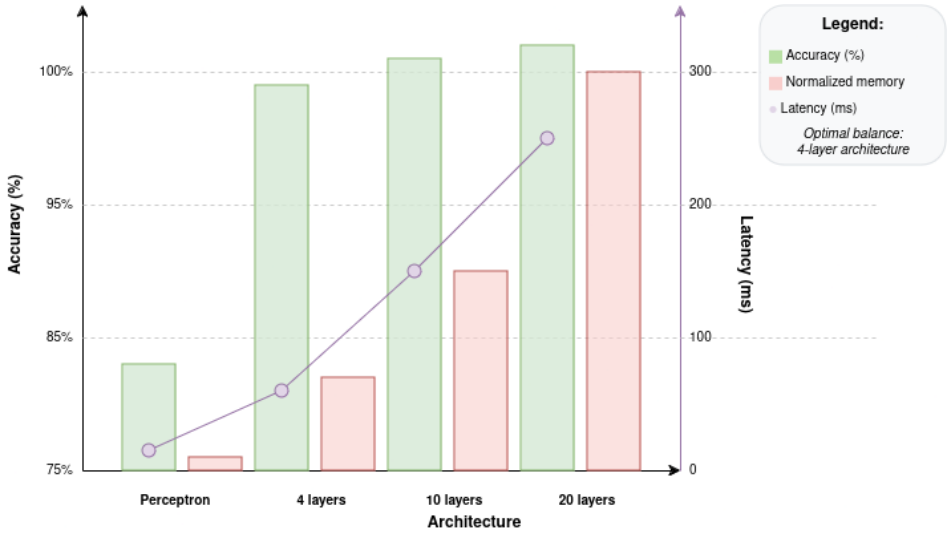

Fig. 3. Representation of the accuracy-latency relationship.

The analysis revealed the existence of three distinct regions:

- High efficiency region (78-90% accuracy): linear resource growth
- Moderate efficiency region (90-95% accuracy): quadratic growth
- Low efficiency region (95%+ accuracy): exponential growth

The optimal operating point is located at the boundary between the first and second regions, corresponding to the four-layer architecture (Figure 3).

CONCLUSION

The comprehensive study of evolutionary integration of neural networks of different architectural complexity in mathematical computing systems made it possible to obtain fundamental results that have significant theoretical and practical importance for the development of a new generation of intelligent computing systems [14].

The developed evolutionary integration methodology provides a scientifically rigorous approach to the systematic implementation of neural networks of progressive complexity. Mathematical formalization of the integration process through multi-criteria efficiency indicators allows evidence-based decision-making about the feasibility of transitioning to more complex architectures [15]. The proposed quantitative criteria comprehensively consider not only classification accuracy, but also resource consumption, reliability, and long-term economic factors.

The experimental study definitively confirmed the existence of a fundamental nonlinear dependence between architectural complexity and the practical effectiveness of neural networks in the context of mathematical computations [16]. Mathematical modeling revealed the logistic nature of the dependence of accuracy on the number of parameters, with a clearly expressed saturation effect at depths over 10 layers.

The single-layer perceptron demonstrated exceptional resource utilization efficiency with an acceptable accuracy of 78.3%, making it the optimal choice for resource-constrained systems and high-throughput applications. Exceptional stability (σ = 0.31%) and minimal resource footprint confirm the feasibility of using the perceptron as the fundamental basis for evolutionary integration.

The four-layer architecture achieved an optimal balance between accuracy (94.1%) and efficiency, representing the best point for most practical applications. Significant accuracy improvement of 15.8% with a moderate increase in resource costs makes this

architecture the preferred choice for balanced systems with moderate performance requirements.

The ten-layer network showed diminishing returns with an accuracy of 95.6% at a substantial increase in resource consumption. The marginal improvement of 1.5% does not justify the sharp increase in operational costs for most applications.

The twenty-layer architecture clearly demonstrates the effect of excessive complexity with minimal improvement to 96.8% at exponential growth in resource requirements [17]. Economic analysis shows a negative return on investment for most commercial applications.

The critical significance of the study lies in the mathematical proof of the existence of optimal architectural complexity that minimizes the total cost of ownership while maximizing practical utility [18]. The established logistic model allows predicting the effectiveness of arbitrary architectures and optimizing system design in early development stages.

The practical significance of the results lies in the possibility of scientifically grounded optimization of mathematical computing systems through informed selection of neural network architecture. The formulated recommendations allow maximizing system efficiency, considering specific constraints and requirements of particular applications [19].

The theoretical contribution of the study includes the development of a comprehensive mathematical framework for analyzing the effectiveness of evolutionary integration of neural networks and establishing fundamental laws governing the relationship between architectural complexity and system performance [20].

The theoretical contribution of the study includes the development of a comprehensive mathematical framework for analyzing the effectiveness of evolutionary integration of neural networks and establishing fundamental laws governing the relationship between architectural complexity and system performance [20].

While this study focused specifically on neural network architectures, future research will expand to include comparative analysis with traditional machine learning methods such as Random Forest, SVM, and ensemble approaches, providing a comprehensive evaluation framework for algorithm selection in mathematical computing systems.

Future research directions include expanding the methodology to alternative architectures (convolutional, recurrent), investigating adaptive algorithms for automatic architecture selection, and developing hybrid approaches that combine the advantages of different model types for specialized mathematical computing applications.

ACKNOWLEDGMENTS AND FUNDING SOURCES

The authors are grateful for the support from the Ministry of Education and Science of Ukraine (Project No 0125U001883).

COMPLIANCE WITH ETHICAL STANDARDS

The authors declare that the research was conducted in the absence of any potential conflict of interest.

AUTHOR CONTRIBUTIONS

Conceptualization, [O.K.]; methodology, [O.K]; investigation, [M.B.]; software, [M.B.]; data curation, [M.B.]; writing – original draft preparation, [M.B.]; writing – review and editing, [O.K].

All authors have read and agreed to the published version of the manuscript.

REFERENCES

[1] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444. https://doi.org/10.1038/nature14539

[2] Stipsitz, M., Sanchis-Alepuz, H. Approximating the steady-state temperature of 3D electronic systems with convolutional neural networks. Mathematical and Computational Applications, 2022, 27(1), 7. https://doi.org/10.3390/mca27010007
[3] Kang, S., Jung, J., Lee, S. A hybrid deep learning model predicting pressure distribution images for 2D airfoils from coordinate information. Nature Scientific Reports, 2024, 14, 30764. https://doi.org/10.1038/s41598-024-84940-w
[4] Bolesta, I., Kushnir, O., Bavdys, M., Khvyshchun, I., Demchuk, A. Computational-Measurement System "Nanoplasmonics". Part 1: Architecture. 2019 XIth International Scientific and Practical Conference on Electronics and Information Technologies (ELIT). IEEE, 2019, pp. 164-167. https://doi.org/10.1109/ELIT.2019.8892288
[5] Bolesta, I., Kushnir, O., Bavdys, M., Khvyshchun, I., Demchuk, A. Computational-Measurement System "Nanoplasmonics". Part 2: Structure of Microservices. 2019 XIth International Scientific and Practical Conference on Electronics and Information Technologies (ELIT). IEEE, 2019, pp. 172-175. https://doi.org/10.1109/ELIT.2019.8892345
[6] Li, N., Ma, L., Xing, T., Yu, G., Wang, C., Wen, Y., Cheng, S., Gao, S. Automatic design of machine learning via evolutionary computation: A survey. Applied Soft Computing, 2023, 143, 110412. https://doi.org/10.1016/j.asoc.2023.110412
[7] Ying, H., Song, M., Tang, Y., Xiao, S., Xiao, Z. Enhancing deep neural network training efficiency and performance through linear prediction. Scientific Reports, 2024, 14(1), 15027. https://doi.org/10.1038/s41598-024-65691-0
[8] Cuomo, S., Di Cola, V.S., Giampaolo, F., Rozza, G., Raissi, M., Piccialli, F. Scientific machine learning through physics-informed neural networks: Where we are and what's next. Journal of Scientific Computing, 2022, 92(3), 88. https://doi.org/10.1007/s10915-022-01939-z
[9] Duan, S., Yu, S., Principe, J. Modularizing deep learning via pairwise learning with kernels. IEEE Transactions on Neural Networks and Learning Systems, ArXiv.. https://doi.org/10.48550/arXiv.2005.05541
[10] Raissi, M., Perdikaris, P., Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 2019, 378, 686-707. https://doi.org/10.1016/j.jcp.2018.10.045
[11] Beck, C., Weinan, E., Jentzen, A. Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations. Journal of Nonlinear Science, 2019, 29(4), 1563-1619. https://doi.org/10.1007/s00332-018-9525-3
[12] De Ryck, T., Mishra, S. Generic bounds on the approximation error for physics-informed (and) operator learning. Advances in Neural Information Processing Systems, 2022, 35, 10945-10958. https://doi.org/10.48550/arXiv.2205.11393
[13] Lu, L., Jin, P., Pang, G., Zhang, Z., Karniadakis, G.E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nature Machine Intelligence, 2021, 3(3), 218-229. https://doi.org/10.1038/s42256-021-00302-5
[14] Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L. Physics-informed machine learning. Nature Reviews Physics, 2021, 3(6), 422-440. https://doi.org/10.1038/s42254-021-00314-5
[15] Mishra, S., Molinaro, R. Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for PDEs. IMA Journal of Numerical Analysis, 2022, 42(2), 981-1022. https://doi.org/10.1093/imanum/drab032
[16] Berner, J., Grohs, P., Jentzen, A. Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of Black–Scholes partial differential equations. SIAM Journal on Mathematics of Data Science, 2020, 2(3), 631-657. https://doi.org/10.1137/19M125649X
[17] Haghighat, E., Raissi, M., Moure, A., Gomez, H., Juanes, R. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. Computer

Methods in Applied Mechanics and Engineering, 2021, 379, 113741. https://doi.org/10.1016/j.cma.2021.113741
[18] Beck, C., Becker, S., Grohs, P., Jaafari, N., Jentzen, A. Solving the Kolmogorov PDE by means of deep learning. Journal of Scientific Computing, 2021, 88(3), 73. https://doi.org/10.1007/s10915-021-01590-0
[19] Sirignano, J., Spiliopoulos, K. DGM: A deep learning algorithm for solving partial differential equations. Journal of Computational Physics, 2018, 375, 1339-1364. https://doi.org/10.1016/j.jcp.2018.08.029
[20] Bengio, Y., Lodi, A., Prouvost, A. Machine learning for combinatorial optimization: a methodological tour d'horizon. European Journal of Operational Research, 2021, 290(2), 405-421. https://doi.org/10.1016/j.ejor.2020.07.063
[21] Rathi, N., Chakraborty, I., Kosta, A., Sengupta, A., Ankit, A., Panda, P., Roy, K. Exploring neuromorphic computing based on spiking neural networks: Algorithms to hardware. ACM Computing Surveys, 2023, 55(12), 1-49. https://doi.org/10.1145/3571155
[22] Hu, Q., Zhang, H., Gao, F., Xing, C., An, J. Analysis on the number of linear regions of piecewise linear neural networks. IEEE Transactions on Neural Networks and Learning Systems, 2022, 33(2), 644-653. https://doi.org/10.1109/TNNLS.2020.3028431

ПОЕТАПНА ІНТЕГРАЦІЯ НЕЙРОННИХ МЕРЕЖ РІЗНОЇ АРХІТЕКТУРИ В СИСТЕМАХ МАТЕМАТИЧНИХ ОБЧИСЛЕНЬ

Михайло Бавдис  , ***Олексій Кушнір*** 
Кафедра радіофізики та комп'ютерних технологій,
Львівський національний університет імені Івана Франка
вул. Ген. Тарнавського, 107, 79017, м. Львів, Україна

АНОТАЦІЯ

Вступ. Розвиток сучасних систем математичних обчислень потребує ефективного впровадження алгоритмів машинного навчання з урахуванням балансу між точністю прогнозування та обчислювальними ресурсами. Особливої уваги заслуговує питання поетапної інтеграції нейронних мереж різної складності з мінімізацією ризиків для виробничих систем та дослідження ефекту насичення при збільшенні глибини архітектури.

Матеріали та методи. Метою дослідження є розробка методології еволюційної інтеграції нейронних мереж у системах математичних обчислень з порівняльним аналізом чотирьох архітектур: перцептрон, багатошарова мережа (4 шари), глибока мережа (10 шарів) та надглибока архітектура (20 шарів). Експерименти проводилися на наборі даних з 45 000 зразків та 24 характеристиками.

Результати. Виявлено логістичний характер залежності точності від складності архітектури. Перцептрон показав базову точність, багатошарова мережа покращила результат на 15.8%, глибока мережа - на 1.5%, надглибока - лише на 1.2%. Ресурсні витрати зростали експоненційно зі збільшенням складності архітектури.

Висновки. Розроблено методологію поетапного впровадження нейронних мереж з урахуванням ефекту насичення точності. Результати показують оптимальність багатошарових архітектур для більшості практичних задач. Виявлений ефект насичення дозволяє приймати обґрунтовані рішення щодо архітектурного вибору в ресурсно-обмежених системах

Ключові слова: Нейронні мережі, еволюційна інтеграція, математичні обчислення, глибоке навчання, ефект насичення, оптимізація архітектури.

Received / Одержано	Revised / Доопрацьовано	Accepted / Прийнято	Published / Опубліковано
05 December, 2025	12 January, 2026	20 January, 2026	30 March , 2026