Journal homepage: https://itce.vn.ua/en

Information Technologies and Computer Engineering **Vol. 23, No. 2. 2026**

Article's History: Received: 19.01.2026 Revised: 20.05.2026
Accepted: 25.06.2026 Published: 03.08.2026

ISSN 1999-9941; e-ISSN 2078-6387
UDC 004.85
DOI: 10.31649/vitce/2.2026.100

Logarithmic transformation of input activations in convolutional network neurons: An approach and analysis of effectiveness using MNIST classification as an example

Artem Tarnovskyi*
Postgraduate Student
Vinnytsia National Technical University
21021, 95 Khmelnytske Highway Str., Vinnytsia, Ukraine
https://orcid.org/0009-0006-0811-8611

Abstract. Convolutional neural networks attract considerable attention because they can learn directly from input data, automatically forming the features required for classification. The purpose of the study was to improve the architecture of a convolutional neural network by applying a logarithmic transformation of input activations in the neurons of convolutional layers and examining its influence on classification accuracy and the robustness of the neural network to changes in image brightness. The mathematical model of a modified neuron was considered; it evaluates the input signal through a logarithmic transformation of the form $\gamma \cdot ln(x+1)$, where γ is an additional trainable parameter that determines the degree of nonlinear signal compression. This approach increased the sensitivity of the neuron to relative changes in input signals, reduces the influence of large activation values, and can improve the stability of the training process. An analysis of gradients for the trainable parameters of the neural network was conducted within the backpropagation method. The analysis indicated that logarithmic transformation changes the pattern of weight updates, strengthening the gradient at small signal values and weakening it at large values. Particular attention was paid to the γ parameter, which acts as a global scale coefficient for the neural layer and enables the network to adjust the level of nonlinearity independently. Experimental verification of the effect of logarithmic transformation of input activations in convolutional layers was conducted using the MNIST handwritten digit classification task with a LeNet-like architecture. The experimental results demonstrated that, for standard images, the logarithmic model provides only a slight increase in accuracy compared with the model with classical linear neurons, averaging 99.17% versus 99.11%. However, for the same images with reduced brightness, the logarithmic model demonstrated high robustness and much higher accuracy. In particular, when brightness was reduced sevenfold, classification accuracy for the model with classical neurons decreased on average to 27.11%, whereas the logarithmic model maintained a level of approximately 90.98%. The results confirmed the appropriateness of using logarithmic transformation as a mechanism for improving the stability of neural networks in computer vision tasks under insufficient or variable illumination

Keywords: logarithmic neuron; LeNet-like network; convolutional layer; image classification; deep learning; gradient descent

Introduction

Deep learning has become one of the key areas of artificial intelligence and machine learning. This is particularly relevant amid increasing requirements for the robustness of deep learning models when working with data affected by substantial variations in illumination and noise. Contemporary computer vision tasks also often involve processing data with a wide dynamic range, which complicates the effective training of models using only linear representations. In this context, the search for new methods of nonlinear transformation of input signals directly within neuron structures is a promising area for improving both the accuracy and the generalisation ability of models. L. Alzubaidi *et al*. (2021) presented a comprehensive analysis of the evolution of deep learning (DL). The researchers emphasised the advantage of DL approaches over conventional machine learning because they can automatically

Suggested Citation:
Tarnovskyi, A. (2026). Logarithmic transformation of input activations in convolutional network neurons: An approach and analysis of effectiveness using MNIST classification as an example. *Information Technologies and Computer Engineering*, 23(2), 100-108. doi: 10.31649/vitce/2.2026.100.

*Corresponding author (*tarnovskiy0211@gmail.com*)

extract features from input data, which is critical for many applied tasks.

Convolutional neural networks are among the most widely used neural networks in deep learning because they automatically obtain structural representations from data. A. Younesi *et al*. (2024) noted that convolution operations in deep learning serve as powerful feature extractors, enabling neural networks to learn effectively from raw data. This makes convolutional networks a highly effective tool for computer vision tasks, such as object identification and image classification. R. Raj & A. Kos (2025) indicated that convolutional networks are evolving rapidly and that their architectures are becoming increasingly complex and diverse. The researchers demonstrated that advances in large-scale computing and the use of graphics processing unit resources enable the creation of massive multilayer architectures. S. Khan & R. Iqbal (2025) stated that most architectural innovations from 2015 to 2025 were associated with increasing network depth and using the spatial characteristics of data more effectively. The researchers outlined that greater depth enables a network to obtain more diverse features, but does not guarantee improved learning ability. Therefore, increasing the width of layers, that is, using more filters per layer, is preferred to further increasing the number of layers. Studies from 2015 to 2025 are also largely associated with innovations in convolution operations, pooling strategies, normalisation methods, and regularisation methods. P. Tsirtsakis *et al*. (2025) established that, alongside the development of new models, algorithms, and tools, substantial efforts and resources are directed towards improving dataset quality. Since the quality of the training dataset directly affects model performance, datasets that better reflect real-world conditions are important. Nevertheless, the main principles of signal processing in convolutional networks remain almost unchanged: they are mainly linear operations limited by threshold activations such as ReLU. However, real-world data often have a nonlinear character, where the information value of a feature can increase disproportionately to its magnitude. For example, according to the Weber-Fechner law (Maes, 2021), the intensity of human perception changes in proportion to the logarithm of stimulus strength. Accordingly, applying a logarithmic transformation to input signals enables a neuron to perceive relative changes in input activations as absolute increments. Since the growth of the logarithmic function slows as its argument increases, logarithmic transformation can reduce the magnitude of gradients for activations with large absolute values, which may help stabilise the training process. Therefore, applying a logarithmic transformation to input signals directly inside a neuron can enable a neural network to automatically compress the value range of these signals and focus on their relative changes.

Despite these potential advantages, this approach is applied either in a limited form or to solve other tasks. B.A. Maxwell *et al*. (2024) examined the application of logarithmic transformation to input RGB images as a means of increasing the robustness of convolutional neural networks to changes in illumination and colour balance. The experiments showed that ResNet18 and DenseNet121 networks trained on logarithmised RGB data performed better on all versions of the test set, namely the original set, the set with altered colour balance, the set with altered intensity, and the set with both augmentations, than networks trained on linear data. Other approaches use a logarithmic transformation to represent input neuron activations as powers of two, which reduces data bit width and replaces multiplication with a shift operation. One example of this approach was described by H. Madadum & Y. Becerikli (2022), who presented a modified logarithmic quantisation method. In particular, specialised logarithmic quantisation of input activations is used; it is based on representing their values as powers of two using fixed-point numbers. The proposed approach compresses models to 4-6 bits without a substantial loss of accuracy and without the need for further training.

Despite the existence of individual studies, the application of logarithmic transformation is limited to input data preprocessing or computational optimisation tasks, whereas its integration directly into the computational elements of neural networks remains insufficiently studied. In particular, there is no systematic analysis of the influence of such transformation on the training process and the generalisation properties of models. The purpose of the study is to improve the architecture of a convolutional neural network by applying a logarithmic transformation of input activations in the neurons of convolutional layers and examining its influence on classification accuracy and the robustness of the neural network to changes in image brightness.

Materials and Methods

The mathematical model of a neuron that evaluates the input signal through logarithmic transformation has the following form:

$$y = f\left(\sum_{i=1}^{n} w_i \cdot ln\left(x_i + \varepsilon\right) + b\right), \quad (1)$$

where y is the neuron output; f is the activation function; x_i are input signals; ε is a constant greater than 0; w_i are input weights; b is the bias, which shifts the activation function graph for better adaptation of the model. In this study, the neuron (1) is designated as logarithmic. Adding ε ensures a positive non-zero value under the logarithm. When working with images, x_i are pixel brightness values, so for first-layer neurons, $x_i \geqslant 0$, and, accordingly, $x_i + \varepsilon > 0$. Since ReLU is used as the activation function f, this inequality also holds for neurons in other layers. The value $\varepsilon = 1$ is adopted to address negative logarithm values for $x_i \in (0, 1)$. Accordingly, expression (2) takes the following form:

$$y = f\left(\sum_{i=1}^{n} w_i \cdot ln(x_i + 1) + b\right). \quad (2)$$

The function $ln(x_i + 1)$ is positive for any $x_i \geqslant 0$. When x_i is zero, the value of $ln(x_i + 1)$ is also zero. This maps a zero input signal to a zero output. For infinitesimal x_i, the

derivative of $ln(x_i + 1)$ is approximately equal to 1, so the use of logarithmic transformation does not cause gradient explosion when x_i is close to zero.

Pixel value normalisation through scaling to the range [0, 1] is often used to accelerate neural network training convergence (Stenin *et al.*, 2022). With respect to transformation (2) for first-layer neurons, the function $ln(x_i + 1)$ transforms the range [0, 1] into the range [0, 0.693], meaning that the degree of compression is quite substantial, especially for x_i values close to 1, which correspond to brighter pixels. The base of the logarithm is reduced to decrease the degree of compression by selecting it as $\exp(1/\gamma)$, where $\gamma > 1$. Considering this, the final mathematical model of the logarithmic neuron has the following form:

$$y = f(\sum_{i=1}^{n} w_i \cdot log_{e^{1/\gamma}}(x_i + 1) + b) = f(\sum_{i=1}^{n} w_i \cdot \gamma \cdot ln(x_i + 1) + b). \quad (3)$$

The product $w_i \cdot \gamma$ can be considered an ordinary weight of the i-th neuron input with a different value, which the network could obtain by modifying w_i during training, because γ is absorbed by w_i; however, this is not the case in the proposed model. The γ parameter is an additional trainable parameter shared by all neurons in the layer. It acts equally on all logarithmically transformed neuron inputs and controls the form of nonlinear compression. It is assumed that, during training, the network would use this parameter to adapt the steepness of the logarithmic curve independently by changing the degree of compression of the input activation range of neurons. Thus, γ is considered a scale parameter.

The derivative of the logarithm $ln(x_i + 1)$, which is equal to $1/(x_i + 1)$, increases as x_i decreases. This means that, for the input layer, neurons (3) are more sensitive to changes in brightness in dark pixels than in bright pixels. This may be useful under low image brightness or low contrast. In this case, $\gamma > 1$ can further increase contrast and extract important details from shadows. Thus, logarithmic transformation of the form $\gamma \cdot ln(x_i + 1)$ can be expected to provide the network with the ability to focus on structural differences created by contrasts and contours.

The possible effect of applying a logarithmic transformation of input activations is examined using the classification of handwritten digits from the Modified National Institute of Standards and Technology (MNIST) dataset. The MNIST dataset is widely used in machine learning and consists of 60,000 28 × 28 grey-scale images of handwritten digits for training and 10,000 similar images for testing (Wang *et al.*, 2020). A convolutional LeNet-like neural network is used for the study. The LeNet architecture is well-suited to low-resolution images and, because of its relatively small number of parameters, provides an optimal balance between accuracy and computational cost. More modern but more complex convolutional network architectures, such as ResNet or VGG, can provide higher classification accuracy (Zhe, 2024), but they are designed for processing images with greater information capacity and were therefore considered excessive for this task.

The classical LeNet architecture contains two convolutional layers based on filters with 5 × 5 kernels. Modern approaches rely on increasing the number of convolutional layers and using filters with smaller kernels, particularly 3 × 3, which provide higher classification accuracy than 5 × 5 or 7 × 7 filters (Khanday *et al.*, 2021; Nocentini *et al.*, 2022). The main and most widely used activation function is ReLU. However, the question of the advantages of small filters remains open. M. Aboukhair *et al.* (2025) noted that preference for small filters, such as 3 × 3 filters, is biased and that the potential of large filters has not been fully assessed. Considering this, the architecture of the LeNet-like convolutional network used in the study was determined through the experiments conducted in this analysis (Fig. 1). The main task addressed when defining the network architecture is to obtain a classification accuracy of at least 99% with a comparatively small number of trainable parameters.

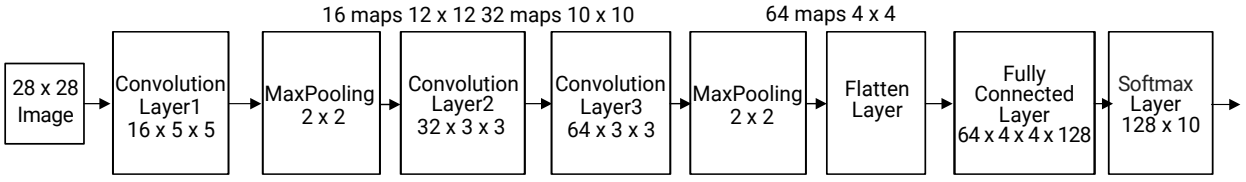

Figure 1. Proposed LeNet-like eight-layer architecture for MNIST classification
Source: developed by the researcher

The proposed LeNet-like network consists of three convolutional layers with ReLU activation functions, two MaxPooling subsampling layers, a flatten layer, one fully connected layer also with a ReLU activation function, and a Softmax output layer for classification. The first convolutional layer consists of 16 filters with a 5 × 5 kernel (the number of trainable parameters is 16 × (5 × 5 + 1) = 416). The second convolutional layer consists of 32 filters with a 3 × 3 kernel for each of the 16 channels of the previous layer (the number of trainable parameters is 32 × (16 × 3 × 3 + 1) = 4,640). The third convolutional layer consists of 64 filters with a 3 × 3 kernel for each of the 32 channels of the previous layer (the number of trainable parameters is 64 × (32 × 3 × 3 + 1) = 18,496). After the third convolutional layer, 2 × 2 pooling is used, which produces 64 feature maps of size 4 × 4 at the input to the fully connected layer. The fully connected layer consists of 128 neurons (the number of trainable parameters is 128 × (64 × 4 × 4 + 1) = 131,200). The Softmax output layer contains 10 neurons (the number of trainable parameters

is 10 × (128 + 1) = 1,290). Thus, the total number of network parameters is 156,042.

A software application using the PyTorch Python framework and the Python 3.10 programming language was created to build the neural network with the proposed architecture, train it, and test it. Data visualisation in the form of graphs was implemented using the Matplotlib library. The network was trained using stochastic gradient descent (SGD). The cross-entropy function was used as the loss function. The learning rate was set to 0.1. During training, L2 regularisation with a coefficient of 0.005 was applied to prevent overfitting. Dropout with a parameter of 0.25 was applied in the second and third convolutional layers. The number of epochs was 30. The MNIST training set of 60,000 images was divided into training and validation samples of 50,000 and 10,000 samples, respectively, to assess neural network model quality during training. For each epoch, the training sample was shuffled, while the order of samples in the validation set remained unchanged. A random number generator with a fixed seed was used to ensure result stability. The batch sizes for training and validation were set to 64. The study was conducted for two networks with the architecture shown in Figure 1: one with classical linear neurons (the baseline model) and one with logarithmic neurons (3) in the convolutional layers (the logarithmic model). The training conditions for both networks were completely identical: training and validation in each epoch were performed on the same batches

Results and Discussion

The training results for the baseline model are presented in Figure 2. The obtained graphs illustrate that the error on the training data decreased rapidly from 0.6 to 0.1 by approximately the third epoch. The error then decreased slowly throughout the rest of the training to approximately 0.02. This indicates that the model learned well. The error on the validation data also initially decreased rapidly from 0.5 to 0.1 in approximately one epoch and then gradually decreased to approximately 0.05. However, fairly large upward fluctuations in error were observed between epochs 12 and 15 and near epoch 25. This demonstrates a certain instability in the training process and some signs of overfitting. Accuracy on the validation sample quickly reached 0.98 near epoch 4 and then gradually increased with minor fluctuations around 0.985 and periodic decreases. The best accuracy on the validation data, 98.94%, was obtained at iteration 22,678.

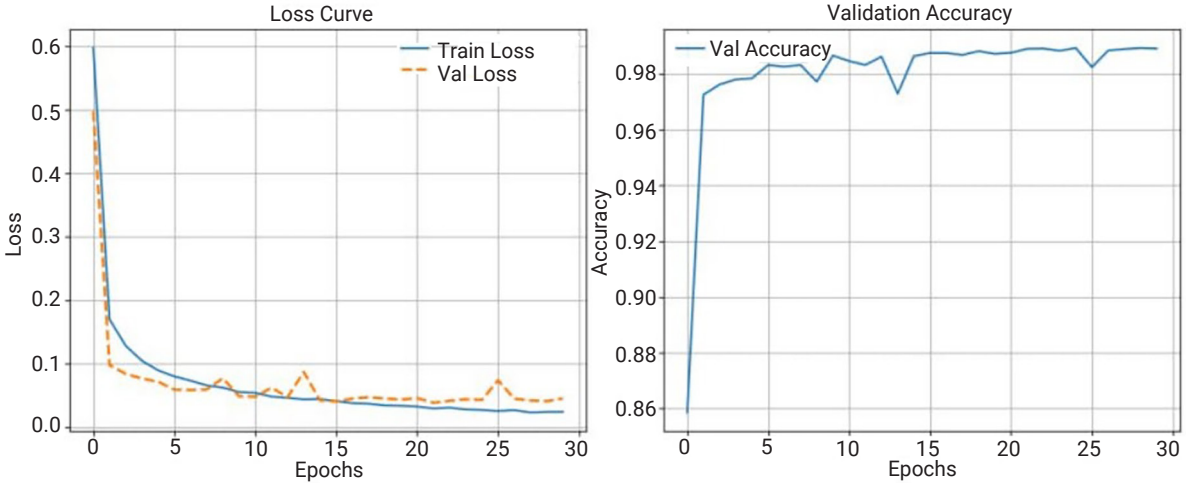

Figure 2. Changes in error and accuracy during training for the neural network with classical linear neurons
Source: obtained during the experiment using Python 3.10 and Matplotlib

Figure 3 presents the training results for the model with logarithmic transformation of input activations. Compared with the baseline model, the logarithmic model demonstrated more stable behaviour during training. The error on the training data changed almost in the same way as in the baseline model. The error on the validation data decreased from approximately 0.22 to 0.05 by epoch 10 and remained almost unchanged throughout all subsequent epochs. Substantial fluctuations in validation error occurred only until epoch 6. Accordingly, accuracy on the validation sample changed substantially in the initial epochs, reaching 0.985 by approximately epoch 6. Accuracy then gradually increased with minor fluctuations to a level above 0.99, which was reached at approximately epoch 22. The best accuracy on the validation data, 99.07%, was obtained at iteration 23,460.

The behaviour of the trainable scale parameter γ was also analysed; in the software application, it was represented by the trainable parameter named log_scale. Figure 4 presents the dynamics of change in this parameter during training for each of the three logarithmic convolutional layers. For each layer, the scale γ was initialised with Euler's number ($e \approx 2.718$). Figure 4 shows that the behaviour of γ depended on layer depth. For the first convolutional layer, γ immediately became greater than 3 and remained almost unchanged throughout all epochs. For the second and third layers, γ decreased monotonically, although the rate of decrease gradually slowed, reaching approximately

1.8 in the second layer and approximately 1.7 in the third. This indicates that the network automatically adapted the degree of logarithmic transformation depending on the level of feature abstraction. In deeper layers, input activations were more strongly compressed in amplitude, reducing the influence of large values. The results confirm that, during training, the network regulated the scale of logarithmic transformation in each layer.

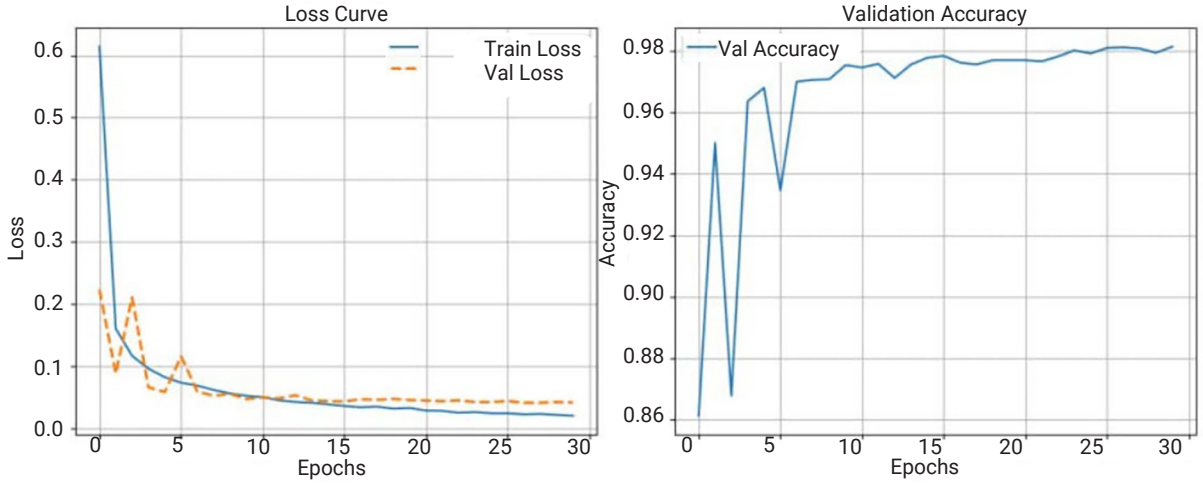

Figure 3. Changes in error and accuracy during training for the neural network with logarithmic neurons in convolutional layers

Source: obtained during the experiment using Python 3.10 and Matplotlib

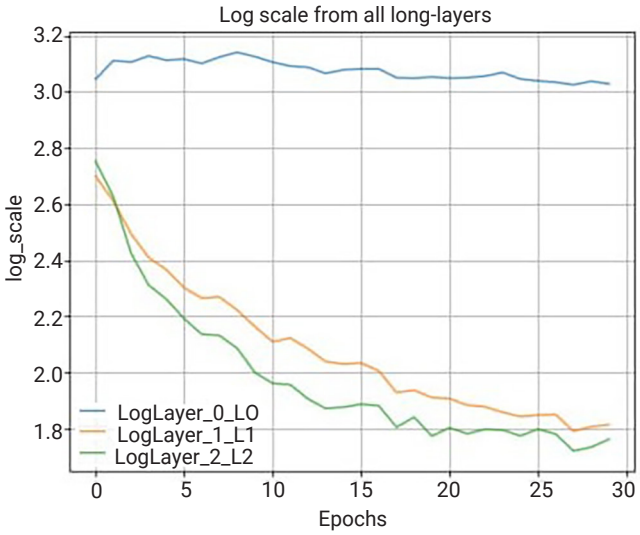

Figure 4. Change in the scale γ (log_scale) during training for each convolutional layer

Source: obtained during the experiment using Python 3.10 and Matplotlib

Classification accuracy on the MNIST test dataset was 99.14% for the baseline model and 99.21% for the neural network with convolutional layers using logarithmic neurons. The logarithmic transformation of input activations had its strongest effect when working with low-brightness images. When the brightness of test images was reduced threefold, classification accuracy was 97.91% for the baseline model and 98.92% for the logarithmic model. With a fivefold reduction in brightness, the results were 87.20% and 97.08%, respectively. For images whose brightness was reduced sevenfold, the baseline model demonstrated classification accuracy of 37.63%, whereas the logarithmic model achieved 87.57%. Testing on reduced-brightness images was conducted without retraining the models, meaning that the models had been trained on unchanged images from the MNIST training set. During testing, the networks were presented with unchanged images from the MNIST test set and the same images with reduced brightness: first threefold, then fivefold, and then sevenfold. This approach assesses the ability of the model to generalise training results to images of different brightness levels. Notably, final classification accuracy is determined by training quality, which depends in part on the distribution of samples among training batches. A series of further experiments

with different seed values was conducted to obtain other random distributions of the training sample and increase the representativeness of the results. The experimental results are presented in Table 1.

Table 1. Test results for the neural network with classical linear neurons and the network with logarithmic neurons in convolutional layers

No.	MNIST classification accuracy, %							
	Original brightness		Brightness reduced ×3		Brightness reduced ×5		Brightness reduced ×7	
	Base model	Log model	Base model	Log model	Base model	Log model	Base model	Log model
1	99.14	99.21	97.91	98.92	87.20	97.08	37.63	87.57
2	99.28	99.26	90.65	98.89	32.55	97.43	12.78	90.65
3	99.21	99.33	96.27	99.01	30.13	97.90	11.41	94.45
4	99.03	99.18	83.45	98.18	17.44	91.53	11.35	72.81
5	99.05	99.03	97.77	98.60	76.82	96.60	24.56	89.43
6	99.01	99.14	95.31	98.94	48.79	98.05	14.35	96.64
7	99.16	99.19	97.99	99.00	88.03	98.20	50.98	96.34
8	99.06	99.05	93.02	98.79	27.05	97.44	11.37	93.31
9	98.97	99.10	97.89	98.71	86.09	97.34	44.93	93.77
10	99.15	99.22	98.58	99.06	91.72	98.33	51.70	94.79
Mean value	99.11	99.17	94.88	98.81	58.58	96.99	27.11	90.98

Source: compiled by the researcher based on the experimental results

On unchanged images, both models showed approximately the same results: mean accuracy was 99.11% for the baseline model and 99.17% for the logarithmic model. Thus, the use of logarithmic transformation of input activations in the neurons of convolutional layers only slightly increases the classification accuracy of baseline MNIST images. The best result was an accuracy improvement of 0.15% (result no. 4): accuracy on the test images was 99.18% for the logarithmic model versus 99.03% for the baseline model. For some distributions of the training sample, classification accuracy on the test sample was even lower for the logarithmic model (results no. 2, no. 5, and no. 8). When image brightness decreased, the advantage of the logarithmic model became more pronounced. With a threefold reduction in brightness ($k = 3$), mean accuracy was 98.81% for the logarithmic model versus 94.88% for the baseline model; for $k = 5$, it was 96.99% versus 58.58%; and for $k = 7$, it was 90.98% versus 27.11%. Thus, the mean accuracy increase of the logarithmic model was 3.93%, 38.41%, and 63.87%, respectively. The advantage of the logarithmic model was observed not only in mean values but also in stability: even when the logarithmic model showed lower accuracy on standard MNIST images, it outperformed the baseline model at all three dimming levels. This indicates that applying a logarithmic transformation of input activations in the neurons of convolutional layers makes the network robust to changes in input image brightness.

The results demonstrate that the proposed architecture of the convolutional LeNet-like neural network provides high MNIST classification accuracy with a comparatively small number of parameters, while logarithmic transformation of input activations in the neurons of convolutional layers gives it high robustness to decreases in input image brightness. The baseline model, which contains 156,042 parameters, showed a mean classification accuracy of 99.11%. The logarithmic model with the same architecture, which has three more trainable parameters because of the logarithmic scale parameter in the convolutional layers, demonstrated a mean accuracy of 99.17%. For comparison, S. Kadam *et al*. (2020) obtained one of the best MNIST classification results, 99.29%, using an architecture with four convolutional layers based on filters with 3×3 kernels and a fully connected layer with 512 neurons. The total number of parameters in that model was 594,922, which exceeds the complexity of the proposed architecture by more than 3.8 times. Under sevenfold dimming, the accuracy of the baseline model decreased to 27.11%, whereas the logarithmic model showed a reduction in accuracy to 90.98%, confirming its high invariance to changes in signal intensity. Similar results, namely the sensitivity of a classical convolutional network to brightness changes and robustness when logarithmic transformation of input data is applied, were presented by M. Cotogni & C. Cusano (2022). The researchers indicated that the transition from linear RGB space to logarithmic space transforms the multiplicative relationship between pixel colour and illumination intensity into an additive offset. The researchers implemented a mathematically complex extension for modifying standard components of the neural network architecture and transforming it into an offset-equivariant network, namely a network whose outputs receive the same increments as the input signals. As a result, identical changes in input signals have the same effect on network outputs without changing its internal logic. This makes the network robust to changes in illumination. In contrast to this approach, the logarithmic transformation of input activations proposed in this study achieves such robustness in a simpler way.

Network invariance to illumination is a critical factor in applications where images are obtained under variable or insufficient lighting conditions. C. Hu *et al*. (2023)

examined the effect of changes in input image brightness on the classification accuracy of modern deep neural network architectures, such as DenseNet, ResNet50, VGG19, GoogleNet, MobileNet, and AlexNet. The results obtained by the researchers emphasised that the standard architectures selected for the study demonstrate a substantial decrease in classification accuracy when brightness changes. Using ResNet50, which achieved the best results in all experiments, the researchers confirmed that data augmentation methods, such as AugMix, for expanding the training sample, together with a transition to architectures without batch normalisation and without pretraining, increase the robustness of classification results to changes in image brightness. However, augmentation methods only partially address the robustness problem because they can expand the statistical distribution of the training sample, but do not change the way the network processes data. If brightness changes differently in real-world conditions than during augmentation, the model may show low accuracy.

The loss of accuracy when image brightness decreases can be minimised not only through modification of the internal architecture of the neural network or supplementation of the training sample with images whose brightness is randomly changed, but also through preliminary correction of input images to restore their informativeness. Within this approach, J.A. Rodríguez-Rodríguez *et al*. (2026) proposed an adaptive system for selecting image enhancement methods to support stable classification accuracy of a convolutional network when input data brightness decreases. The system uses a pretrained regressor that analyses the characteristics of input images and predicts the change in cross-entropy of the convolutional network to select the optimal processing method from Gamma Correction, Histogram Equalization, Contrast Limited Adaptive Histogram Equalization, and Logarithmic Transformation. The results of experiments with AlexNet and GoogLeNet networks using the ILSVRC2012 dataset showed that the best classification accuracy under extremely low input image brightness was provided by Logarithmic Transformation, which is consistent with the results of this study.

The comparison of the approach proposed in this study with the results of T. Weidler *et al*. (2021) and J. Petkovic & R. Fioresi (2024) is of particular interest because those researchers used architectural solutions based on the lateral inhibition mechanism associated with the functioning of retinal photoreceptors. T. Weidler *et al*. (2021) implemented this mechanism by introducing interaction between output channels of a convolutional layer, namely Semantic Lateral Connectivity (SemLC). According to this approach, each channel is strengthened by close neighbours and suppressed by more distant filters. During training, filters that recognise similar features self-organise and group according to their semantic proximity. The proposed architectural solution was examined in classification tasks on the CIFAR-10 and MNIST datasets. Three simplified modifications of AlexNet were used for the first dataset, and the CapsNet capsule neural network was used for the second. The introduction of SemLC improved accuracy in all experiments, although the increase was very small. For MNIST, the mean accuracy was 99.58% compared with 99.56% for the baseline CapsNet. For comparison, the logarithmic transformation of input activations considered in this study provided a mean accuracy increase of 0.06%. Although the proposed architecture of the LeNet-like network is inferior in classification accuracy, by an average of 0.42% for baseline variants and 0.39% for modified variants, it requires substantially fewer computational resources: approximately 156 thousand parameters compared with more than 8 million in CapsNet.

J. Petkovic & R. Fioresi (2024) implemented the lateral inhibition mechanism using a so-called precortical module added as the first layer of a convolutional network. The module consists of three consecutive convolutional layers with 7 × 7 filters, a hyperbolic tangent activation function, and a Dropout regularisation mechanism. During training, the filter weights of the module are adjusted independently so that their sum approaches zero, causing the filters to acquire a centre-surround structure. This gives the network the ability to respond to brightness differences rather than to its absolute value. The researchers used the standard LeNet-5 as the convolutional network. According to the results of experiments on the MNIST dataset, the baseline LeNet-5 showed an accuracy of 99.1% for unchanged images, while the model with the precortical module showed 98.8%. Under critical dimming, the accuracy of the baseline model decreased to 11.9%, whereas the modified architecture demonstrated an accuracy of 78.7%. These data are consistent with the results obtained for the model with logarithmic transformation of input activations considered in this study: both approaches demonstrate invariance to changes in brightness. However, the source of this invariance differs: in the model with the precortical module, it is achieved through spatial filtering, where the neuron responds to the brightness difference between the central pixel and its surroundings. With logarithmic transformation of input activations, the neuron response becomes proportional to relative differences in pixel brightness, making it insensitive to absolute brightness values. Notably, the precortical module requires the introduction of three additional convolutional layers, which substantially increases the number of model parameters. By contrast, the proposed logarithmic transformation adds only one parameter to each existing convolutional layer, preserving the high throughput of the network.

Conclusions

Logarithmic transformation of input activations inside a neuron transforms multiplicative changes in signals at its inputs into additive ones, which makes the neuron sensitive to relative signal changes in a manner similar to biological perception principles. This study proposed a modified neuron that performs the transformation $\gamma\ ln(x + 1)$, where γ is a trainable parameter that enables the network to adjust the degree of nonlinear compression of input

activations. From an optimisation perspective, this creates an additional degree of freedom for adjusting neuron sensitivity. The experimental results obtained on the standard MNIST dataset showed that the use of modified neurons in convolutional layers does not reduce classification quality and gives a slight improvement in accuracy, on average from 99.11% to 99.17%. However, the most significant effect is observed when working with dimmed images, where the logarithmic model substantially outperforms the model with classical linear neurons.

Statistical analysis of a series of experiments with different random variations of the training sample confirmed the advantage of logarithmic transformation: the mean increase in accuracy compared with the baseline model was 3.93% under threefold dimming, reaching 63.87% under a sevenfold reduction in brightness. In the latter case, the mean accuracy of the logarithmic model was 90.98%. The results indicate that logarithmic transformation can be an effective tool for increasing the robustness of neural networks to changes in input signal intensity. This makes the neural network with modified neurons more suitable for working with images whose brightness can vary within substantial limits and for applications in computer vision systems operating under low or variable illumination. Further research may focus on analysing model behaviour on more complex datasets and applying this approach in other deep learning architectures.

Acknowledgements
None.

Funding
None.

Conflict of Interest
None.

References

[1] Aboukhair, M., Alsheref, F., Assiri, A., Koura, A., & Kayed, M. (2025). CNN filter sizes, effects, limitations, and challenges: An exploratory study. *Network Computation in Neural Systems*, 1-29. doi: 10.1080/0954898X.2025.2533865.

[2] Alzubaidi, L., Zhang, J., Humaidi, A.J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M.A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, 8(1), article number 53. doi: 10.1186/s40537-021-00444-8.

[3] Cotogni, M., & Cusano, C. (2022). Offset equivariant networks and their applications. *Neurocomputing*, 502, 110-119. doi: 10.1016/j.neucom.2022.06.118.

[4] Hu, C., Shi, W., Li, C., Sun, J., Wang, D., Wu, J., & Tang, G. (2023). Impact of light and shadow on robustness of deep neural networks. *arXiv*. doi: 10.48550/arXiv.2305.14165.

[5] Kadam, S., Adamuthe, A., & Patil, A. (2020). CNN model for image classification on MNIST and fashion-MNIST dataset. *Journal of Scientific Research*, 64 (2), 374-384. doi: 10.37398/JSR.2020.640251.

[6] Khan, S., & Iqbal, R. (2025). A comprehensive survey on architectural advances in deep CNNs: Challenges, applications, and emerging research directions. *arXiv*. doi: 10.48550/arXiv.2503.16546.

[7] Khanday, O.M., Dadvandipour, S., & Lone, M.A. (2021). An effect of filter sizes on image classification in CNN: A case study on CIFAR10 and Fashion-MNIST datasets. *IAES International Journal of Artificial Intelligence*, 10(4), 872-878. doi: 10.11591/ijai.v10.i4.pp872-878.

[8] Madadum, H., & Becerikli, Y. (2022). A resource-efficient convolutional neural network accelerator using fine-grained logarithmic quantization. *Intelligent Automation & Soft Computing*, 33(2), 681-695. doi: 10.32604/iasc.2022.023831.

[9] Maes, C. (2021). Statistical mechanical foundation of Weber-Fechner laws. *Journal of Statistical Physics*, 182, article number 49. doi: 10.1007/s10955-021-02726-0.

[10] Maxwell, B.A., Singhania, S., Patel, A. Kumar, R., Fryling, H., Li, S., Sun, H., He, P., & Li, Z. (2024). Logarithmic lenses: Exploring log RGB data for image classification. In *2024 IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (pp. 17470-17479). Seattle: Institute of Electrical and Electronics Engineers. doi: 10.1109/CVPR52733.2024.01654.

[11] Nocentini, O., Kim, J., Bashir, M.Z., & Cavallo, F. (2022). Image classification using multiple convolutional neural networks on the fashion-MNIST dataset. *Sensors*, 22(23), article number 9544. doi: 10.3390/s22239544.

[12] Petkovic, J., & Fioresi, R. (2024). Spontaneous emergence of robustness to light variation in CNNs with a precortically inspired module. *Neural Computers*, 36(9), 1832-1853. doi: 10.1162/neco_a_01691.

[13] Raj, R., & Kos, A. (2025). An extensive study of convolutional neural networks: Applications in computer vision for improved robotics perceptions. *Sensors*, 25(4), article number 1033. doi: 10.3390/s25041033.

[14] Rodríguez-Rodríguez, J.A., López-Rubio, E., Jiménez-Segura, S., & Molina-Cabello, M.A. (2026). Adaptive image enhancement for robust CNN classification under low illumination. *Expert Systems with Applications*, 315, article number 131696. doi: j.eswa.2026.131696.

[15] Stenin, A., Pasko, V., Soldatova, M., & Drozdovych, I. (2022). Recognition of handwritten numbers based on convolutional neural networks. *Adaptive Automatic Control Systems*, 2(41), 39-44. doi: 10.20535/1560-8956.41.2022.271337.

[16] Tsirtsakis, P., Zacharis, G., Maraslidis, G.S., & Fragulis, G.F. (2025). Deep learning for object recognition: A comprehensive review of models and algorithms. *International Journal of Cognitive Computing in Engineering*, 6, 298-312. doi: 10.1016/j.ijcce.2025.01.004.

[17] Wang, Y., Li, F., Sun, H., Li, W., Zhong, C., Wu, X., Wang, H., & Wang, P. (2020). Improvement of MNIST image recognition based on CNN. In *IOP conference series: Earth and environmental science of 7th annual international conference on geo-spatial knowledge and intelligence 20-21 December 2019, Guangzhou, China* (vol. 428, article number 012097). Guangzhou: Guangzhou Institute of Geography. doi: 10.1088/1755-1315/428/1/012097.

[18] Weidler, T., Lehnen, J., Denman, Q., Sebők, D., Weiss, G., Driessens, K., & Senden, M. (2021). Biologically inspired semantic lateral connectivity for convolutional neural networks. *arXiv*. doi: 10.48550/arXiv.2105.09830.

[19] Younesi, A., Ansari, M., Fazli, M., Ejlali, A., Shafique, M., & Henkel, J. (2024). A comprehensive survey of convolutions in deep learning: Applications, challenges, and future trends. *IEEE Access*, 12, 41180-41218. doi: 10.1109/ACCESS.2024.3376441.

[20] Zhe, W. (2024). Research on the application of CNN in MNIST image classification and recognition. In *2024 IEEE 7th international conference on information systems and computer aided education (ICISCAE)* (pp. 698-706). Dalian: Shenyang Normal University. doi: 10.1109/ICISCAE62304.2024.10761186.

Логарифмічне перетворення вхідних активацій у нейронах згорткових мереж: підхід та аналіз ефективності на прикладі класифікації MNIST

Артем Тарновський

Аспірант
Вінницький національний технічний університет
21021, вул. Хмельницьке шосе, 95, м. Вінниця, Україна
https://orcid.org/0009-0006-0811-8611

Анотація. Згорткові нейронні мережі привертають значну увагу завдяки здатності навчатися безпосередньо на вхідних даних, автоматично формуючи ознаки, необхідні для класифікації. Метою дослідження було вдосконалення архітектури згорткової нейронної мережі за рахунок застосування логарифмічного перетворення вхідних активацій у нейронах згорткових шарів та дослідження його впливу на точність класифікації та стійкість нейромережі до змін яскравості зображень. Розглянуто математичну модель модифікованого нейрона, що оцінює вхідний сигнал через логарифмічне перетворення виду $\gamma \cdot ln(x+1)$, де γ є додатковим навчальним параметром, що визначає ступінь нелінійного стиснення сигналу. Такий підхід дозволив зробити нейрон чутливим до відносних змін вхідних сигналів, зменшити вплив великих значень активацій та потенційно покращити стабільність процесу навчання. Було проведено аналіз градієнтів навчальних параметрів нейронної мережі в межах методу зворотного поширення помилки. Показано, що логарифмічне перетворення змінює характер оновлення ваг, підсилюючи градієнт при малих значеннях сигналу та послаблюючи для великих. Окрему увагу було приділено параметру γ, який виступає як глобальний коефіцієнт масштабу для нейронного шару і дозволяє мережі самостійно налаштовувати рівень нелінійності. Експериментальна перевірка ефекту від логарифмічної трансформації вхідних активацій у згорткових шарах була проведена на задачі класифікації рукописних цифр MNIST із використанням LeNet-подібної архітектури. Результати експериментів показали, що на стандартних зображеннях логарифмічна модель дає лише незначне підвищення точності за порівнянням з моделлю з класичними лінійними нейронами (у середньому 99,17 % проти 99,11 %). Проте для тих самих зображень зі зниженою яскравістю логарифмічна модель продемонструвала високу стійкість та набагато більшу точність. Зокрема, при зменшенні яскравості у 7 разів точність класифікації для моделі з класичними нейронами знижується у середньому до 27,11 %, тоді як логарифмічна модель зберігає рівень близько 90,98 %. Отримані результати підтвердили доцільність використання логарифмічного перетворення як механізму покращення стабільності нейронних мереж при вирішенні задач комп'ютерного зору в умовах недостатнього або змінного освітлення

Ключові слова: логарифмічний нейрон; LeNet подібна мережа; згортковий шар; класифікація зображень; глибоке навчання; градієнтний спуск