Mathematical Problems of Computer Science 60, 63–71, 2023.
doi: 10.51408/1963-0110

UDC 004.93

Convolutional Neural Network (CNN) Layer Development for Effectiveness of Classification Tasks

Rafayel M. Veziryan[1] and Rafayel N. Khachatryan[2]

[1]Institute for Informatics and Automation Problems of NAS RA, Yerevan, Armenia
[2]Questrade Armenia Inc., Yerevan, Armenia
e-mail: rafaelveziryan@gmail.com, raf.khachatryan4@gmail.com

Abstract

This paper presents a novel 2D convolutional layer motivated by the principles of Partial Differential Equation (PDE) of Neural Interaction. Our objective is to leverage this layer to enhance the classification accuracy of Deep Convolutional Neural Networks (DCNN) for various classification tasks. We place a particular emphasis on its integration within the ResNet architecture, and we conduct experimental evaluations on the CIFAR10 and STL10 datasets to validate its efficacy.

Keywords: DCNN, Classification task, Image processing, ResNet, PDE, Cable equation, Grid method.
Article info: Received 30 September 2023; sent for review 2 October 2023; received in revised form 14 November 2023; accepted 16 November 2023.
Acknowledgement: We thank Professor Armenak Babayan for proposing the fundamental problem that guided this research, and for his consistent consultations.

1. Introduction

Deep Learning is a subfield of machine learning [1]. Neural networks simulate the learning process of the human brain. This article explores the synergy between deep learning and PDEs, specifically in the context of enhancing image classification accuracy using ResNet architecture [2].

PDEs hold significant importance in the realms of sound, image, and video processing [3]. Their application in image processing primarily revolves around noise removal and reconstruction[4, 5]. The foundational models relying on PDEs are adept at noise reduction while simultaneously maintaining the image integral features.

Transitioning our focus to DCNNs, challenges like gradient vanishing and gradient exploding have often been obstacles in training Neural Networks effectively. ResNet, or Residual Network, addresses such issues. At its core, the ResNet architecture employs Residual Blocks. These blocks incorporate skip connections, bypassing selected layers, offering a unique approach to handling these challenges. We will delve deeper into the intricacies of the ResNet architecture in the ensuing sections.

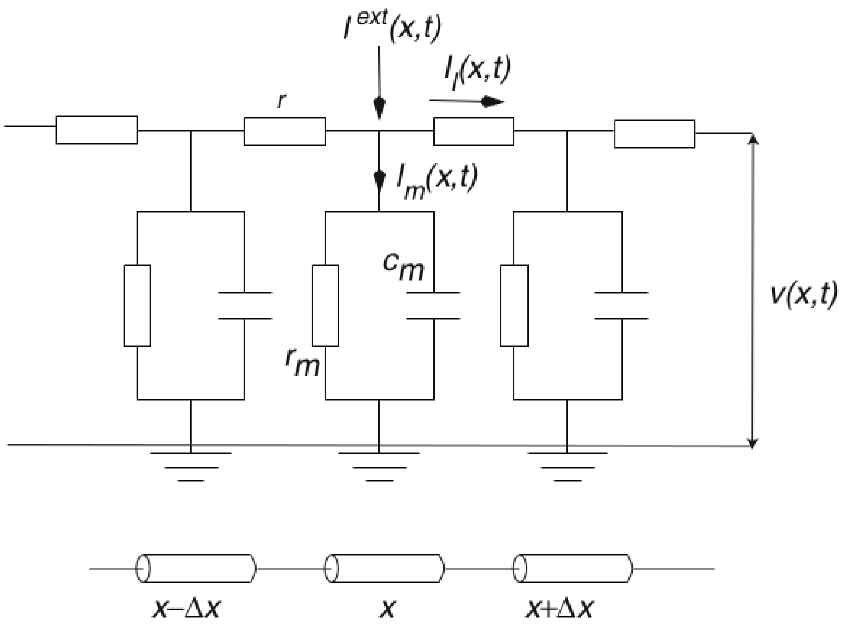

Fig. 1. Equivalent scheme for successive cylindrical segments of a dendritic membrane of a neural cell.

The essence of this article lies in optimizing DCNNs, specifically ResNet, by leveraging the capabilities of PDEs. An insightful source for our research is Bresslof's book, which introduces the Cable Equation [6](Section 1.4). The cable equation describes the variation of potential in neural cells.

Our principal objective is to harness the prowess of PDEs in image processing and integrate it within DCNNs, particularly with ResNet.

2. Mathematical Background

The main focus of this article is on the application of a CNN layer, obtained from Cable equation for classification tasks. As it was mentioned, the cable equation is an important mathematical model of the potential transmission in neural cells. The equivalent scheme is described in Fig. 1.

Taking into account that the transmission of potentials, that is, the transmission of information in neuronal cells, is governed by the equation described above, we propose to use the discretization of this equation in the construction of a neural network. According to the Book designations, the potential transmission in a neuron cell is described by the following equation:

$$\tau_m \frac{\partial v(x,t)}{\partial t} = -v(x,t) + \lambda_m^2 \frac{\partial^2 v(x,t)}{\partial x^2} + r_m I_{ext}(x,t), \ \ t \geq 0, \tag{1}$$

where v is a membrane potential at position x along a cable at time t, C_m is a capacity per unit of the membrane, R is a resistance of the intracellular fluid, R_m is a cell membrane resistance, a is a cable radius, $\tau_m = R_m C_m$ is a membrane time constant and $\lambda_m = (R_m a/2R)^{1/2}$ is a membrane space constant.

Therefore, we propose to apply the discretization of this equation in a neural network. We will employ the grid method to discretize this equation. The grid method is a fundamental technique that transforms PDEs into discrete computational forms. PDEs are converted into algebraic equations by segmenting space into a grid of discrete points, facilitating integration. The distribution of these points establishes the foundation upon which the PDEs are approximated using the finite difference method. The discretized form of the formula is obtained through the finite difference method.

In schematic form, Equation (1) takes the following form:

$$\frac{\partial u}{\partial t} = \alpha \Delta u - u$$

or

$$u_t = \alpha(u_{xx} + u_{yy}) - u.$$

Replace the partial derivatives with their finite difference approximations

$$\frac{u^{t+1}_{i,k} - u^t_{i,k}}{\tau} = \alpha \frac{u^t_{i-1,k} - 2u^t_{i,k} + u^t_{i+1,k}}{h_x^2} + \alpha \frac{u^t_{i,k-1} - 2u^t_{i,k} + u^t_{i,k+1}}{h_y^2} - u^t_{i,k}. \tag{2}$$

From (2) follows

$$u^{t+1}_{i,k} = u^t_{i,k}(1-\tau) + \frac{u^t_{i-1,k} - 2u^t_{i,k} + u^t_{i+1,k}}{\Phi^2} + \frac{u^t_{i,k-1} - 2u^t_{i,k} + u^t_{i,k+1}}{\Psi^2}, \tag{3}$$

where

$$\Phi^2 = \frac{h_x^2}{\alpha \cdot \tau}, \quad \Psi^2 = \frac{h_y^2}{\alpha \cdot \tau}.$$

The resulting scheme is called an explicit scheme because the solution's value at the given moment $t+1$ is strictly obtained by the solution's value in the previous t moment. Equation (3) is equivalent to

$$u^{t+1}_{i,k} = (1-\tau) \cdot u^t_{i,k} + (P_1 + P_2) \cdot U, \tag{4}$$

where

$$U = \begin{bmatrix} u^t_{i-1,k-1} & u^t_{i-1,k} & u^t_{i-1,k+1} \\ u^t_{i,k-1} & u^t_{i,k} & u^t_{i,k+1} \\ u^t_{i+1,k-1} & u^t_{i+1,k} & u^t_{i+1,k+1} \end{bmatrix} sP_1 = \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{\Phi^2} & -\frac{2}{\Phi^2} & \frac{1}{\Phi^2} \\ 0 & 0 & 0 \end{bmatrix} P_2 = \begin{bmatrix} 0 & \frac{1}{\Psi^2} & 0 \\ 0 & -\frac{2}{\Psi^2} & 0 \\ 0 & \frac{1}{\Psi^2} & 0 \end{bmatrix}$$

P_1 and P_2 are defined as two-dimensional, weighted convolution operators for the neural network with weights Φ, Ψ, and τ is also a weight. (3) represents our CNN layer, $u^{t+1}_{i,k}$ is our present layer, and $u^t_{i,k}$ is our previous layer, which we will call a cable equation layer.

3. Architecture

In this paper, we introduce a new component within DCNN, which we call a cable equation layer, and it is designed to be trainable. Our new layer usage highlights the features of the image, and since the layer is trainable, it allows us to optimize its configuration for image processing. Traditional DCNNs begin with a standard convolutional layer. Here we introduce our cable equation layer at the outset, which enables us to process the image before the other layers of the network process it further. We investigate the efficacy of our layer in two convolutional blocks: first, we integrate the cable equation layer into a standard CNN Block, and second, we integrate the cable equation layer into the ResNet Block (see Fig. 2).

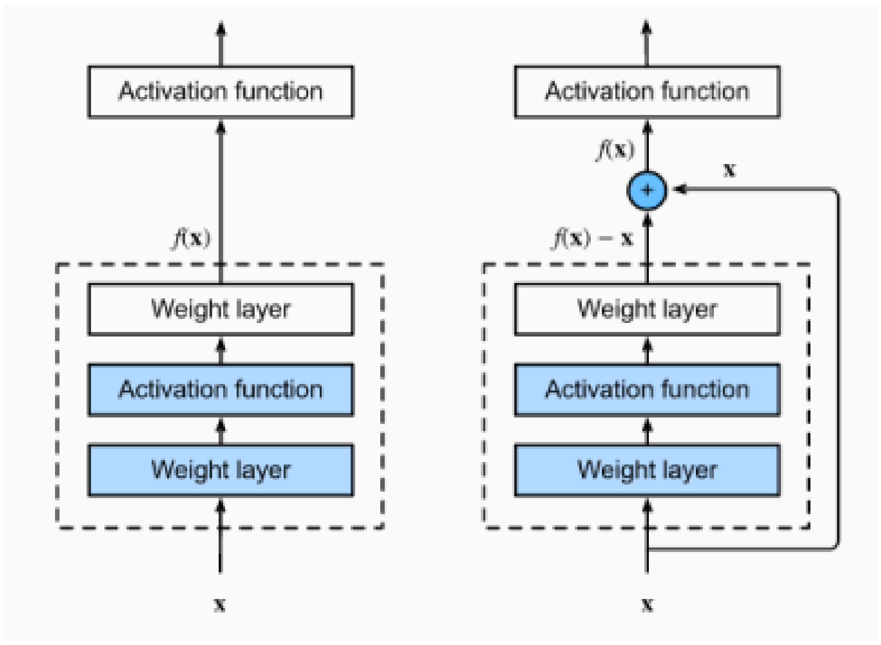

Fig. 2. Left: Standard CNN block, Right: ResNet block.

4. Experiments

The CIFAR10 dataset [7] is used for our experiments, consisting of 50K training images and 10K testing images. Input images have 3 channels and a size of 32×32, and use the STL10 dataset[8], which consists of 5K training images and 8K testing images, and have 3 channels and a size of 96×96. Network architecture will be presented in the table. Our Cable Equation layer incorporates BatchNorm [9] and employs the ReLU activation function. We conducted two experiments incorporating the Cable Equation layer. In the first experiment, we introduced the Cable Equation layer along with the preceding layers before the core Neural Network. For the second experiment, we crafted a residual block infused with a Cable Equation layer, iteratively applied one or more times, strategically positioned prior to the Neural Network. Table 1 below is a description of the architecture of the original and experimental models for the CIFAR10 dataset.

Initially, we implement the k-cable equalization layer, followed by 3x3 convolutions generating 64 output channels. Subsequently, the architecture encompasses 8 ResNet Blocks, partitioned into four sections, each with output channels (64, 128, 256, 512). The sequence continues with 4x4 averaging for CIFAR10 and 12x12 for STL10, a fully connected layer, and culminates with LogSoftmax activation. We used this architecture for both datasets.
For the CIFAR10 dataset, we initiate preprocessing steps before feeding the data into the neural network. This involves padding each side with 4 pixels, followed by a random crop of size 32x32. Additionally, a RandomHorizontalFlip operation is applied, and ultimately,

Table 1. CIFAR10 dataset: From top to bottom: Original ResNet, ResNet with 1, 2, 3 CabEq standard blocks, ResNet with 1, 2, 3 CabEqBl Residual Blocks. O. S. is the output shape after each layer.

Original		$\begin{bmatrix} 3\times 3, \\ 64 \end{bmatrix}$	$\begin{bmatrix} 3\times 3, \\ 64 \\ 3\times 3, \\ 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times 3, \\ 128 \\ 3\times 3, \\ 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times 3, \\ 256 \\ 3\times 3, \\ 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times 3, \\ 512 \\ 3\times 3, \\ 512 \end{bmatrix} \times 2$	Average pooling(4×4), 10d FC, LogSoftmax
CabEqSdBl1	$\begin{bmatrix} 3\times 3, & 3 \end{bmatrix} \times 1$						
CabEqSdBl2	$\begin{bmatrix} 3\times 3, & 3 \end{bmatrix} \times 2$						
CabEqSdBl3	$\begin{bmatrix} 3\times 3, & 3 \end{bmatrix} \times 3$						
CabEqRNBl1	$\begin{bmatrix} 3\times 3, & 3 \\ 3\times 3, & 3 \end{bmatrix} \times 1$						
CabEqRNBl2	$\begin{bmatrix} 3\times 3, & 3 \\ 3\times 3, & 3 \end{bmatrix} \times 2$						
CabEqRNBl3	$\begin{bmatrix} 3\times 3, & 3 \\ 3\times 3, & 3 \end{bmatrix} \times 3$						
O. S.	32×32	32×32	32×32	16×16	8×8	4×4	1×1

the image is normalized using the means of (0.4914, 0.4822, 0.4465) and standard deviations of (0.2023, 0.1994, 0.2010). These transformations are conducted on the training data. For the test data, normalization is exclusively applied using the same means and standard deviations. The outcomes for CIFAR10 are presented in Table 2.

Table 2. The result of testing CIFAR10 dataset in Original ResNet and ResNet with Cable equation standard blocks and Cable equation ResNet blocks.

Model	Accuracy(%)	Parameter cnt
Original	88.92	11181642
CabEqSdBl 1	89.41	11181675
CabEqSdBl 2	89.57	11181708
CabEqSdBl 3	89.4	11181741
CabEqRNBl 1	89.68	11181708
CabEqRNBl 2	89.83	11181774
CabEqRNBl 3	88.9	11181840

For the STL10 dataset, we employ preprocessing procedures tailored to its distinct characteristics. To prepare the data for neural network input, we apply 4-pixel padding to all sides, followed by a randomized crop of dimensions 96x96. Additionally, a RandomHorizontalFlip operation is implemented. Subsequently, the image is normalized using mean values of (0.44671097, 0.4398105, 0.4066468) and standard deviations of (0.2603405, 0.25657743, 0.27126738). These transformations are integral to the training data, while for the test data, normalization is consistently applied using the same mean and standard deviation parameters. Table 3 shows the architecture for STL10. The outcomes for STL10 are presented in Table 4.

Table 3. STL10 dataset: From top to bottom: Original ResNet, ResNet with 1, 2, 3 CabEq layers, ResNet with 1, 2, 3 CabEq Residual Blocks. O. S. is the output shape after each layer.

Original		$\begin{bmatrix}3\times 3,\\ 64\end{bmatrix}$	$\begin{bmatrix}3\times 3,\\ 64\\ 3\times 3,\\ 64\end{bmatrix}\times 2$	$\begin{bmatrix}3\times 3,\\ 128\\ 3\times 3,\\ 128\end{bmatrix}\times 2$	$\begin{bmatrix}3\times 3,\\ 256\\ 3\times 3,\\ 256\end{bmatrix}\times 2$	$\begin{bmatrix}3\times 3,\\ 512\\ 3\times 3,\\ 512\end{bmatrix}\times 2$	Average pooling(12 × 12), 10d FC, LogSoftmax
CabEqSdBl1	$\begin{bmatrix}3\times 3, & 3\end{bmatrix}\times 1$						
CabEqSdBl2	$\begin{bmatrix}3\times 3, & 3\end{bmatrix}\times 2$						
CabEqSdBl3	$\begin{bmatrix}3\times 3, & 3\end{bmatrix}\times 3$						
CabEqRNBl1	$\begin{bmatrix}3\times 3, & 3\\ 3\times 3, & 3\end{bmatrix}\times 1$						
CabEqRNBl2	$\begin{bmatrix}3\times 3, & 3\\ 3\times 3, & 3\end{bmatrix}\times 2$						
CabEqRNBl3	$\begin{bmatrix}3\times 3, & 3\\ 3\times 3, & 3\end{bmatrix}\times 3$						
O. S.	96 × 96	96 × 96	96 × 96	48 × 48	24 × 24	12 × 12	1 × 1

Table 4. The result of testing STL10 dataset in Original ResNet and ResNet with Cable equation standard blocks and Cable equation ResNet blocks.

Model	Accuracy(%)	Parameter cnt
Original	73.625	11181642
CabEqSdBl 1	75.68	11181675
CabEqSdBl 2	75.33	11181708
CabEqSdBl 3	70.8	11181741
CabEqRNBl 1	75.025	11181708
CabEqRNBl 2	75.6625	11181774
CabEqRNBl 3	75.45	11181840

Our training strategy involves stochastic gradient descent (SGD)[10] coupled with a momentum of 0.9. The learning rate is managed using a OneCycleLR scheduler[11], featuring a maximum rate of 0.5. Throughout the training process, we iterate through 100 epochs for both datasets. Fig. 3 and Fig. 4 show the loss and accuracy for testing images per epoch for the CIFAR10 and STL10 datasets.

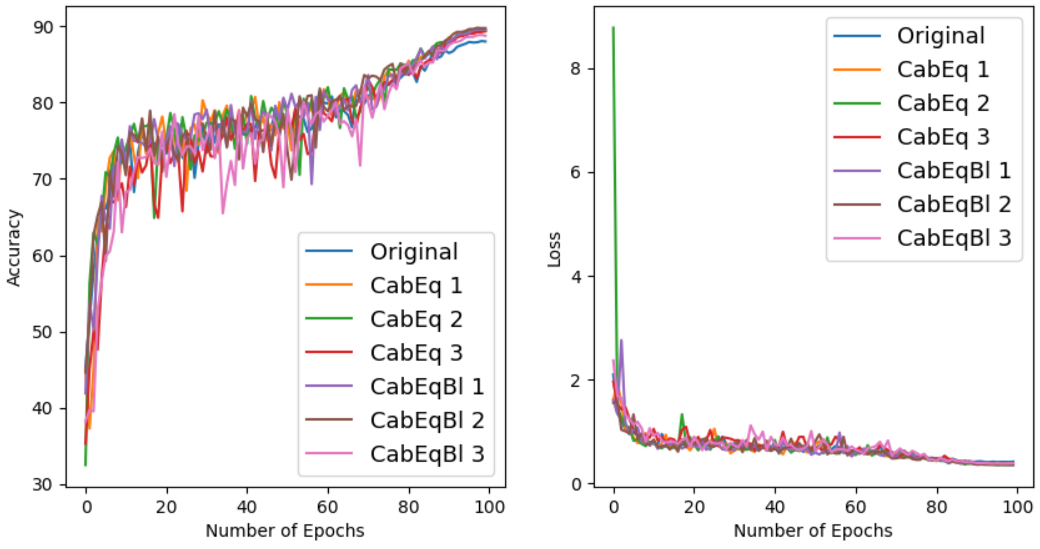

Fig. 3. Left: Relation between accuracy and epoch count for testing images of CIFAR10 dataset, Right: Relation between loss and epoch count for testing images of CIFAR10 dataset

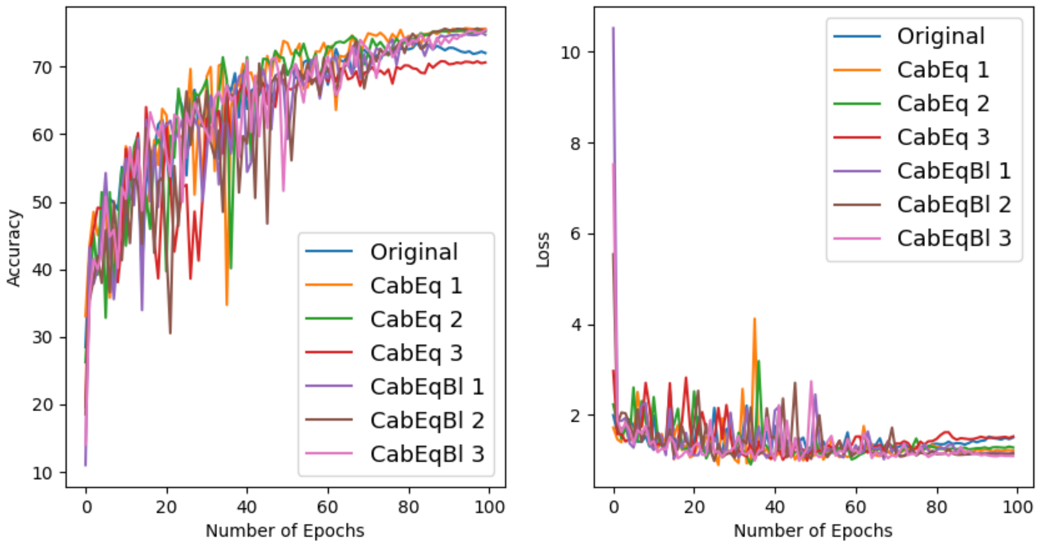

Fig. 4. Left: Relation between accuracy and epoch count for testing images of STL10 dataset, Right: Relation between loss and epoch count for testing images of STL10 dataset

5. Conclusion

The aim of this article is to increase the efficiency of DCNNs for classification tasks. We obtained the discretization of the PDE using the Grid method. Based on this, a convolution layer was constructed from the obtained convolution operators, which we called this the Cable equation layer. Motivated by the application of PDEs in image processing, we built the architecture by adding the Cable equation layer in front of the DCNN as a learnable image processing layer. This enables us to find the best layer to process the image. Our experiments were conducted on the ResNet architecture, and the tests were performed on the CIFAR10 and STL10 datasets. Based on the obtained results, we can say that having a layer with a few parameters can increase effectiveness. The subject of further research can be integrating the Cable equation layer into other architectures, as well as thinking about creating new architectures based on the Cable equation layer itself.

References

[1] I. H. Sarker, "Deep Learning: A comprehensive overview on techniques, taxonomy, applications and tesearch directions", *SN Computer Science*, vol. 2, no. 6, 420, 2021.

[2] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, pp. 770-778, 2016.

[3] K. Mikula, "Image processing with partial differential equations", *Modern Methods in Scientific Computing and Applications. NATO Science Series*, vol. 75, pp. 283-321, 2002.

[4] F. Guichard, L. Moisan and J.-M. Morel, "A review of P.D.E.models in image processing and image analysis", *Journal dePhysique IV (Proceedings)*, vol. 12, no. 1, pp. 137-154, 2002.

[5] L. Ruthotto and E. Haber, "Deep Neural Networks Motivated by Partial Differential Equations", *Journal of Mathematical Imaging and Vision*, vol. 62, no. 3, pp. 352-364, 2020.

[6] P. C. Bressloff, *Waves in Neural Media,* Part of the book series: Lecture Notes on Mathematical Modelling in the Life Sciences, New York, USA, Springer, 2014.

[7] (2023) CIFAR-10 Dataset. [Online] Available: `http://www.cs.toronto.edu/~kriz/cifar.html`

[8] (2023) STL-10 Dataset. [Online] Available: `http://ai.stanford.edu/~acoates/stl10/`

[9] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating deep network training by reducing internal covariate shift", *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France, pp. 448-456, 2015.

[10] S. Ruder, "An overview of gradient descent optimization algorithms", arXiv:1609.04747, 2020.

[11] A. Al-Kababji, F. Bensaali and S. P. Dakua, "Scheduling techniques for liver segmentation: ReduceLRonPlateau vs OneCycleLR", *The 2nd International Conference on Intelligent Systems and Patterns Recognition*, Hammamet, Tunisia, pp. 204-212, 2022.

Կոնվոլյուցիոն նեյրոնային ցանցի շերտի մշակում դասակարգման առաջադրանքների արդյունավետության համար

Ռաֆայել Մ. Վեզիրյան[1] և Ռաֆայել Ն. Խաչատրյան[2]

[1]ՀՀ ԳԱԱ Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտ, Երևան, Հայաստան
[2]Questrade, Երևան, Հայաստան
e-mail: rafaelveziryan@gmail.com, raf.khachatryan4@gmail.com

Ամփոփում

Այս հոդվածը ներկայացնում էնոր 2D կոնվոլյուցիոն շերտ, որը հիմնված էնեյրոնային փոխազդեցության մասնակի դիֆերենցիալ հավասարման սկզբունքների վրա: Մեր նպատակն էօգտագործել այս շերտը՝ բարձրացնելու խորը կոնվոլյուցիոն նեյրոնային ցանցերի դասակարգման ճշգրտությունը տարբեր դասակարգման առաջադրանքների համար: Մենք հատուկ շեշտը դնում ենք ResNet ճարտարապետության մեջ դրա ինտեգրման վրա, և փորձնական գնահատումներ ենք անցկացնում CIFAR10 և STL10 տվյալների հավաքածուների վրա՝ շերտի արդյունավետությունը հաստատելու համար:

Բանալի բառեր՝ Խորը կոնվոլյուցիոն նեյրոնային ցանց, դասակարգման առաջադրանք,պատկերի մշակում, ResNet, մասնակի դիֆերենցիալ հավասարում, մալուխայինհավասարում, ցանցային մեթոդ:

Разработка слоя сверточной нейронной сети для повышения эффективности задач классификации

Рафаэль М. Везирян [1] и Рафаэль Н. Хачатрян[2]

[1]Институт проблем информатики и автоматизации НАН РА, Ереван, Армения
[2]Questrade, Ереван, Армения
e-mail: rafaelveziryan@gmail.com, raf.khachatryan4@gmail.com

Аннотация

В этой статье представлен новый двумерный сверточный слой, основанный на принципах уравнения в частных производных нейронного взаимодействия. Наша цель использовать этот слой для повышения точности классификации глубоких сверточных нейронных сетей для различных задач классификации. Мы уделяем особое внимание его интеграции в архитектуру ResNet и проводим экспериментальные оценки наборов данных CIFAR10 и STL10 для проверки его эффективности.

Ключевые слова: Глубокая сверточная нейронная сеть, задача классификации, обработка изображений, ResNet, уравнение в частных производных, уравнение кабеля, метод сетки.