Improving Convolutional Neural Network Expression via Difference Exponentially Linear Units

Zhiying Hu¹, Hongbo Huang¹,²*, Qinglin Ran³, Mingyang Yuan¹

¹Computer School, Beijing Information Science and Technology University, Beijing, 100192, China
²Institute of Computing Intelligence, Beijing Information Science and Technology University, Beijing, 100192, China
³College of Mechanical and Electrical Engineering, Beijing Information Science and Technology University, Beijing, 100192, China
*Corresponding author’s e-mail: hhb@bistu.edu.cn

Abstract. Convolutional Neural Network (CNN) has been applied to various tasks with great success. Adding an activation function is an important way to introduce nonlinearity into convolutional neural networks. The commonly used activation functions mostly use different forms of negative feedback for the negative input, however, some researchers have recently proposed positive feedback methods for the negative input such as Concatenated Rectified Linear Units (CReLU) and Linearly Scaled Hyperbolic Tangent (LiSHT) and achieved better performance. To explore this idea further more, we propose a new activation function called Difference Exponentially Linear Unit (DELU). The proposed DELU activation function can optionally provide positive and negative feedback for different values of negative inputs. Our experimental results on the commonly used datasets such as Fashion Mnist, CIFAR10, and Imagenet show that DELU outperforms other six activation functions, Leaky ReLU, ReLU, ELU, SELU, Swish, SERLU.

1. Introduction

Convolutional neural network (CNN) has been applied in many fields and achieved many encouraging results. Activation function provides nonlinearity for the convolutional neural network, enabling the CNN to be deeper. The most popular activation function for convolutional neural networks (CNNs) is the Rectified Linear Unit (ReLU)¹. In addition to generating sparsity, the main advantage of ReLU is that they can alleviate the vanishing gradient problem. After that, there are many ReLU-based variants, such as Leaky ReLU², Rrelu³, Prelu⁴, GELU⁵, Elu⁶, SELU⁷, ABRelu⁸. These activations can be used to boost the results in some specific tasks, but the robustness and the stability are pretty poor to some extent. The Swish⁹ activation function proposed by Prajit Ramachandran et al. has shown excellent results in neural networks. Recently, the Scaled Exponentially-Regularized Linear Units (SERLU)¹⁰ proposed by Zhang et al. is no longer monotonous for negative inputs while preserving the self normalizing property, and has shown a stronger effect in their experiments.

Although the aforementioned activation functions have good effect for their simplicity and nonlinearity, they do not provide good feedbacks for negative input signals. ReLU sets all negative inputs to zero, that easily lead to dead neurons in CNN models. The Leaky ReLU, which is a variant of ReLU, converts the negative input into a negative signal output. This adjustment eliminates the
phenomenon of neuron death. CReLU\textsuperscript{[10]} and LiSHT\textsuperscript{[11]} improves ReLU by giving positive feedback to the negative input information in the neural network, and achieved good results in their experiments. Although CReLU and LiSHT are able to give positive feedback through negative signals, they all give up negative feedback. So for different negative input values, it is worth considering to give positive feedback and negative feedback respectively.

In this work, inspired by the characteristics of the SERLU, we propose a new activation for neural networks called Difference Exponentially Linear Unit (DELU). The advantage of DELU is that both positive feedback and negative feedback are respectively given by different negative signals in the convolutional neural network, so that the neural network can better express the characteristic information. In our experiments, we compared with multiple popular activation functions with different network models on several datasets. The results show that DELU outperforms all other activation functions.

The rest of this paper is organized as follows. The second section briefly introduces the work related to the DELU proposed in this article. The third section describes the details of the method we propose. The fourth section introduces the experimental results and analysis.

2. Related work

In this section, we will detail several activation functions and related methods of DELU. In convolutional neural networks community, commonly used activation functions include Rectified linear unit (ReLU), scaled exponential linear unit (SELU), exponential linear unit (ELU), Leaky rectified linear unit (Leaky ReLU) and Swish etc. Recently, Scaled Exponentially-Regularized Linear Units (SERLUs) was proposed and achieved good performance in many scenarios.

Rectified linear unit (ReLU) is the most popular activation in convolutional neural networks. The function is defined as \( y_i = \begin{cases} x_i & \text{if } x_i \geq 0 \\ 0 & \text{if } x_i < 0 \end{cases} \) (1).

Obviously, the function is very simple but shows good robustness in many tasks. That’s the main reason that it is currently the most widely used activation in CNN. However, in the process of back propagation, it is easy to cause neuron death when the input is negative. To avoid the dead neuron phenomenon, many researchers have made various improvements based on ReLU and proposed many new activation functions.

Leaky ReLU (LReLU) modified the output of negative input by a real number factor. The function is defined as \( y_i = \begin{cases} x_i & \text{if } x_i \geq 0 \\ \alpha x_i & \text{if } x_i < 0 \end{cases} \) (2) where \( \alpha \) is a multiplier factor, empirically set to 0.01. Parametric Rectified Linear Unit (PReLU) and Randomized Leaky Rectified Linear Unit (RReLU) are similar to Leaky ReLU. The parameter \( \alpha \) of RReLU is set to a random number and the parameter \( \alpha \) of PReLU is set to be a learnable number. In our comparison experiments we pick LReLU as the representative of these variants.

Recently Zhang et al. proposed Scaled Exponentially-Regularized Linear Units (SERLU). It lead to a bump-shaped function for negative input (See Figure 2), and thus permits the negative input to give positive feedback or negative feedback according to different values. The SERLU activation is as follows (3).

\[ y_i = \lambda \begin{cases} x_i & \text{if } x_i \geq 0 \\ \alpha x_i e^{x_i} & \text{if } x_i < 0 \end{cases} \]

where \( \lambda \approx 1.07862, \alpha \approx 2.90427 \).

All of them convert the negative input signal into other forms of negative signals for output by nonlinear transformation. On the contrary, CReLU and LiSHT convert the negative input signal into a positive signal in different ways and achieved better results. LiSHT converts all negative input signals into positive feedback signals. The formula of LiSHT is given by (4).
\[ y_i = x_i \cdot \tanh(x_i) = x_i \cdot \frac{e^{x_i} - e^{-x_i}}{e^{x_i} + e^{-x_i}} \quad (4) \]

CReLU is not an activation function but an activation scheme. By making an identical copy of the convolved linear responses after convolution, negating them, concatenate both parts of activation, and then apply ReLU altogether, so that it retains both positive and negative phase information while maintaining non-saturation nonlinearity.

3. Difference Exponentially Linear Units (DELU)

In convolutional neural networks the inputs of activations are the output of convolution operations. It is obviously that the negative input should play important role during training. And we assume that simultaneously utilize positive and negative feedback for different values of the input will be more helpful. The same policy is adopt by SERLU. Under this assumption, we designed an new activation function called Difference Exponentially Linear Unit (DELU) as (8).

\[ y_i = \begin{cases} x_i & \text{if } x_i \geq 0 \\ \alpha_1 (x_i \cdot e^{x_i} - \alpha_2 x_i \cdot e^{\alpha_2 x_i}) & \text{if } x_i < 0 \end{cases} \quad (5) \]

The hyperparameter \( \alpha_1 \) of DELU controls the value to which a DELU saturates for negative inputs and the hyperparameter \( \alpha_2 \) of DELU mainly controls the distance between the intersection of the DELU and the negative axis of x-axis, where \( \alpha_2 \in (0, 1) \). The value of \( \alpha_1 \) is recommended to not exceed 2.0, which will easily lead to a gradient explosion during the back-propagation. We have tried to set parameters to be trainable, but the result deviated from our original intention and could not help the network to better converge. We finally leave the parameters to fixed values.

3.1. Ablation study

In order to select a more suitable set of parameters, we conduct an ablation study by evaluation on the Cifar-10 dataset. The experimental process is the same as that in section 4.1. We select 5 values for \( \alpha_1 \) and \( \alpha_2 \) respectively, \( \alpha_1 \in \{2.0, 1.5, 1.0, 0.7, 0.3\} \), \( \alpha_2 \in \{0.9, 0.7, 0.5, 0.3, 0.1\} \). A total of 25 experiments were performed, and each experiment trained 100 epochs. The learning rate is set to 0.001 in order to make the network converge quickly. The result is shown in figure 1.

![Figure 1](image.png)

Figure 1. The abscissa represents the values of parameters \( \alpha_1, \alpha_2 \), and the ordinate represents the average accuracy rate of the last 50 training epochs.

According to the experiments, we got a set of basic parameters that achieved better performance, where \( \alpha_1 = 0.3, \alpha_2 = 0.1 \). It is worth noting that this set of basic parameters is not the best because simple experiments have been performed. Unlike other activation functions, DELU can provide both positive and negative feedback when the inputs are negative. We illustrated the DELU curves with three representative activation functions, as shown in figure 2.
4. Experiments with DELU

To evaluate our proposed DELU activation, we conducted experiments on the Fashion Mnist\textsuperscript{[13]}, Cifar10\textsuperscript{[14]} and Imagenet\textsuperscript{[15]} datasets respectively, with comparison to currently popular activation. We did not use any pre-processing and image enhancement on the datasets, but directly used the original image for training. We chose Tensorflow\textsuperscript{[16]} framework and the Adam\textsuperscript{[19]} optimizer in all experiments. Then we performed model evaluations on the testing set of each datasets after every training epoch. We set the momentum to 0.9 and the batch size to 100.

4.1. CIFAR-10

CIFAR-10 is a dataset consisting of 50,000 training images and 10,000 test images. It contains 10 categories. Because the data distribution is simple, we use a simpler model structure. The network structure we used for training has 5 layers.

| Model     | Val Acc. (%) |
|-----------|--------------|
| ReLU      | 69.08        |
| LReLU     | 69.21        |
| ELU       | 68.59        |
| SELU      | 68.19        |
| Swish     | 66.66        |
| SERLU     | 64.99        |
| **DELU**  | **69.29**    |

Table 1 shows that DELU achieved the highest accuracy on the CIFAR-10 validation set. Overall, the performance of DELU is stronger on CIFAR-10.

4.2. Fashion-Mnist

Fashion-Mnist is a dataset consisting of 60,000 training images and 10,000 test images. It contains 10 categories. The network structure we use is similar to 4.1. The size and number of kernels are the same, only the input and output are different. In this experiment, we trained 200 epochs, and set the learning...
rate to 0.001, decreasing by 0.5 every 50 epochs.

Table 2. Fashion Mnist validation accuracy.

| Model | Val Acc. (%) |
|-------|--------------|
| ReLU  | 91.5         |
| LReLU | 90.94        |
| ELU   | 90.94        |
| SELU  | 90.32        |
| Swish | 91.68        |
| SERLU | 91.87        |
| DELU  | 91.92        |

Table 2 shows that DELU achieved the highest accuracy on the validation set. Overall, the performance of DELU is stronger on Fashion Mnist.

4.3. Imagenet

In order to further test the classification performance of our proposed DELU in convolutional neural networks, we chose to conduct experimental tests on Imagenet. Imagenet 2012. The “residual” introduced in Resnet\textsuperscript{[17]} enables the convolutional neural network to reach deeper and better learn sample features, which is of great significance. Resnet was widely used after it was proposed, and many network architectures used the residual idea of Resnet. In order to get the results faster, we used standard Resnet-18 for training instead of deeper networks. The structure of Resnet-18 is shown in figure 3. We set the batch size to 100 for a total of 20 iterations of training. The initial learning rate is set to 0.01 the first 10 epochs of training, reduced it to 0.001 for the last 10 epochs of training.

![Resnet-18 structure.](image)

Figure 3. Resnet-18 structure. Activation functions and batch normalization\textsuperscript{[18]} are used after each convolutional layer and fully connected layer.

Table 3. Imagenet validation accuracy.

| Model | Top-1 Acc. (%) | Top-5 Acc. (%) |
|-------|---------------|---------------|
| ReLU  | 58.68         | 81.83         |
| LReLU | 58.76         | 81.78         |
| ELU   | 58.85         | 81.87         |
| SELU  | 56.53         | 80.39         |
| Swish | 58.12         | 81.21         |
| SERLU | 55.98         | 80.04         |
| DELU  | 59.20         | 81.94         |

We validated on the validation set after each epoch of training, and select the highest set of validation results obtained in the last three rounds for comparison. Table 3 shows that DELU achieved the highest accuracy on the Imagenet validation set.
4.4. Analysis
Experiments showed that DELU can help the convolutional neural network converge faster and achieve better generalization performance in various tasks. From a formula point of view, DELU needs to perform more exponential operations, which requires a little bit computation time than other activation functions. By comparing the training convergence speed and generalization performance, on the whole, our proposed DELU performs better.

5. Conclusion
In this work, we have introduced a new activation function as Difference Exponentially Linear Units (DELU) to improve convolutional neural networks expressiveness. DELU activation function optionally provides positive and negative feedback for different values of negative inputs. On different datasets, Experiments show that DELU can perform the best results so far. Although DELU activation function will spend a little more computing time, it can help the convolutional neural network converge faster and achieve good generalization ability with the same amount of training. In addition, our future work will be to find a more suitable set of parameter values for DELU to promote the expression of convolutional neural networks.

References
[1] Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. Journal of Machine Learning Research, 15, 315-323.
[2] Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013, June). Rectifier nonlinearities improve neural network acoustic models. In Proc. icml (Vol. 30, No. 1, p. 3).
[3] Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:1505.00853.
[4] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE international conference on computer vision (pp. 1026-1034).
[5] Hendrycks, D., & Gimpel, K. (2016). Bridging nonlinearities and stochastic regularizers with gaussian error linear units. https://www.researchgate.net/publication/.
[6] Clevert, D. A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289.
[7] Klambauer, G., Unterthiner, T., Mayr, A., & Hochreiter, S. (2017). Self-normalizing neural networks. In Advances in neural information processing systems (pp. 971-980).
[8] Dubey, S. R., & Chakraborty, S. (2018). Average biased ReLU based CNN descriptor for improved face retrieval. arXiv preprint arXiv:1804.02051.
[9] Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. arXiv preprint arXiv:1710.05941.
[10] Zhang, G., & Li, H. (2018). Effectiveness of scaled exponentially-regularized linear units (SERLUs). arXiv preprint arXiv:1807.10117.
[11] Shang, W., Sohn, K., Almeida, D., & Lee, H. (2016, June). Understanding and improving convolutional neural networks via concatenated rectified linear units. In international conference on machine learning (pp. 2217-2225).
[12] Roy, S. K., Manna, S., Dubey, S. R., & Chaudhuri, B. B. (2019). LiSHT: Non-parametric linearly scaled hyperbolic tangent activation function for neural networks. arXiv preprint arXiv:1901.05894.
[13] Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747.
[14] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images. Handbook of Systemic Autoimmune Diseases, 1(4).
[15] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Berg, A. C. (2015). Imagenet large scale visual recognition challenge. International journal of computer vision, 115(3), 211-252.

[16] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Ghemawat, S. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467.

[17] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[18] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

[19] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.