Optimization Method of Residual Networks of Residual Networks for Image Classification

Long Lin[1], Hao Yuan[1], Liru Guo[2], Yingqun Kuang[3], and Ke Zhang[2(✉)]

[1] Power Systems Artificial Intelligence Joint Laboratory of SGCC, Global Energy Interconnection Research Institute Co., Ltd, Beijing 102209, China
{linlong, yuanhao}@geiri.sgcc.com.cn

[2] North China Electric Power University, Baoding 071000, Hebei, China
glr9292@126.com, zhangke41616@126.com

[3] Power Supply Service Center, State Grid Hunan Electric Power Company, Changsha 410004, China
blacktigerking@sina.com

Abstract. The activation of a Deep Convolutional Neural Network that overlooks the diversity of datasets has been restricting its development in image classification. In this paper, we propose a Residual Networks of Residual Networks (RoR) optimization method. Firstly, three activation functions (RELU, ELU and PELU) are applied to RoR and can provide more effective optimization methods for different datasets; Secondly, we added a drop-path to avoid over-fitting and widened RoR adding filters to avoid gradient vanish. Our networks achieved good classification accuracy in CIFAR-10/100 datasets, and the best test errors were 3.52% and 19.07% on CIFAR-10/100, respectively. The experiments prove that the RoR network optimization method can improve network performance, and effectively restrain the vanishing/exploding gradients.

Keywords: Image classification · RELU · Parametric exponential linear unit
Exponential linear unit · Residual networks of residual networks
Activation function

1 Introduction

In the past five years, deep learning [1] has made gratifying achievements in various computer vision tasks [2, 3]. With the rapid development of deep learning and Convolutional Neural Networks (CNNs), image classification has bidden farewell to coarse feature problems of manual extraction, and turned it into a new process. Especially, after AlexNet [4] won the champion ship of the 2012 Large Scale Visual Recognition Challenge (ILSVRC) [5], CNNs become deeper and continue to achieve better and better performance on different tasks of computer vision tasks.

To overcome degradation problems, a residual learning framework named Residual Networks (ResNets) were developed [8] to ease networks training, which achieved excellent results on the ImageNet test set. Since then, current state-of-the-art image classification systems are predominantly variants of ResNets. Residual networks of

Residual networks (RoR) [13] adds level-wise shortcut connections upon original residual networks to promote the learning capability of residual networks. The rectified linear unit (RELU) [16] has been adopted by most of the convolution neural networks. RELUs are non-negative; therefore, they have a mean activation larger than zero, which would cause a bias shift for units. Furthermore, the selection of the activation function in the current DCNN model does not take into account the difference between the datasets. Different image datasets are different in variety and quality of image. The unified activation function limits the performance of image classification.

In order to effectively solve the above problem, this paper proposes an RoR network optimization method. To begin with, we analyze the characteristics of the activation function (RELU, ELU and PELU) and construct an RoR network with them. Thus, an RoR optimization based on different datasets is proposed. In addition, analysis of the characteristics of RoR networks suggest two modest mechanisms, stochastic depth and RoR-WRN, to further increase the accuracy of image classification. Finally, through massive experiments on CIFAR datasets, our optimized RoR model achieves excellent results on these datasets.

2 Related Work

Since AlexNet acquired a celebrated victory at the ImageNet competition in 2012, an increasing number of deeper and deeper Convolutional Neural Networks emerged, such as the 19-layer VGG [6] and 22-layer GoogleNet [7]. However, very deep CNNs also introduce new challenges: degradation problems, vanishing gradients in backward propagation and overfitting [15].

In order to overcome the degradation problem, a residual learning frame-work known as ResNets [8] was presented by the authors at the 2015 ILSVRC & COCO 2015 competitions and achieved excellent results in combination with the ImageNet test set. Since then, a series of optimized models based on ResNets have emerged, which became part of the Residual-Networks Family. Huang and Sun et al. [10] proposed a drop-path method, the stochastic depth residual networks (SD), which randomly drops a subset of layers and bypasses them with identity mapping for every mini-batch. To tackle the problem of diminishing feature reuse, wide residual networks (WRN) [11] was introduced by decreasing depth and increasing width of residual networks. Residual networks of Residual networks (RoR) [13] adds level-wise shortcut connections upon original residual networks to promote the learning capability of residual networks, that once achieved state-of-the-art results on CIFAR-10 and CIFAR-100 [12]. Each layer of DenseNet [14] is directly connected to every other layer in a feed-forward fashion. PyramidNet [27] gradually increases the feature map dimension at all units to involve as many locations as possible. ResNeXt [26] exposes a new dimension called cardinality (the size of the set of transformations), as a sential factor in addition to the dimensions of depth and width.

Even though non-saturated RELU has interesting properties, such as sparsity and non-contracting first-order derivative, its non-differentiability at the origin and zero gradient for negative arguments can hurt back-propagation [17]. Moreover, its non-negativity induces bias shift causing oscillations and impeded learning. Since the advent of the well-known RELU, many have tried to further improve the performance of the networks with more elaborate functions. Exponential linear unit (ELU) [17], defined as identity for positive arguments and $\exp(x) - 1$ for negative ones, deals with both increased variance and bias shift problems. Parametric ELU (PELU) [18], an adaptive activation function, defines parameters controlling different aspects of the function and proposes learning them with gradient descent during training.

3 Methodology

In this section, three activation functions (RELU, ELU and PELU) are applied to RoR and can provide more effective optimization methods for different datasets; Secondly, we added a drop-path to avoid over-fitting and widened RoR adding filters to avoid gradient vanish.

3.1 Comparative Analysis of RELU, ELU and PELU

The characteristics and performance of several commonly activation functions (RELU, ELU and PELU) are compared and analyzed as follows.

RELU is defined as:

$$f = \begin{cases} h & \mathit{if} \quad h \geq 0 \\ 0 & \mathit{if} \quad h < 0 \end{cases} \tag{1}$$

It can be seen that RELU is saturated at $h < 0$. Since the derivative of $h \geq 0$ is 1, RELU can keep the gradient from attenuation when $h > 0$, thus alleviating the problem of vanishing gradients. However, RELU outputs are non-negative, so the mean of the outputs will be greater than 0. Learning causes a bias shift for units in next layer. The more the units are correlated, the more serious the bias shift. the higher their bias shift.

ELU is defined as:

$$f = \begin{cases} h & \mathit{if} \quad h \geq 0 \\ \alpha\,(\exp(h) - 1) & \mathit{if} \quad h < 0 \end{cases} \tag{2}$$

The ELU incorporates Sigmoid and ReLU with left soft saturation. The ELU hyperparameter controls the value to which an ELU saturates for negative net inputs. ELUs diminish the vanishing gradients effect as RELUs do. By using a saturated negative part, the CNNs can no longer have arbitrary large negative outputs, which reduces variance. ELU outputs negative values for negative arguments, the network can push the mean activation toward zero, which reduces the bias shift.

PELU can be defined as follows:

$$f = \begin{cases} \frac{a}{b} h & \mathit{if} \quad h \geq 0 \\ a\,(\exp(h/b) - 1) & \mathit{if} \quad h < 0 \end{cases}, a, b > 0 \tag{3}$$

With the parameterization in the PELU function, a and b adjust the characteristics of the exponential function in the negative half axis and control the size of exponential decay and saturation point. a and b can also can adjust the slope of the linear function, to keep the differentiability. The parameters in PELU are updated at the same time as the parameters in the network weight layers during back-propagation.

3.2 RoR Networks with RELU, ELU and PELU

RoR [13] is based on a hypothesis: The residual mapping of residual mapping is easier to optimize than original residual mapping. To enhance the optimization ability of residual networks, RoR can optimize the residual mapping of residual mapping by adding shortcuts level by level based on residual networks.

Figure 2 in [13] shows the RoR (ReLU) architecture. The optimal model is 3-level RoR in [13]. Therefore we adopted 3-level RoR (RoR-3) as our basic architecture in experiments.

RoR-3 includes $3n$ final residual blocks, 3 middle-level residual blocks, and a root-level residual block, among which a middle-level residual block is composed of n final residual blocks and a middle-level shortcut, the root-level residual block is composed of 3 middle-level residual blocks and a root-level shortcut. The projection shortcut is done by 1×1 convolutions. RoR (ReLU) adopts a conv-BN-ReLU order in residual blocks.

For the saturation advantage of ELU and PELU, we designed new RoR architectures by adopting ELU and PELU, as shown in Fig. 1. The sequence of layers in residual block is Conv-PELU/ELU-Conv-BN. The batch normalization (BN) layer reduces the exploding gradient problem. We use 16, 32, and 64 convolutional filters sequentially in the convolutional layers of the three residual block groups, as shown in Fig. 1. Other architectures are the same as RoR (RELU)'s.

It can be seen from the previous section that the output of the ReLU is equal to or greater than 0, which makes the RoR network generate the bias shift when training. The bias shift directly limits the image classification performance of the RoR network. ELU and PELU outputs negative values for negative arguments, which allows the RoR network to push the mean activation toward zero. This reduces the bias shift; thus, performance of RoR is improved. Furthermore, as ELU and PELU saturate as the input gets negatively larger, the neurons can-not have arbitrary large positive and negative outputs and still keep a proper weighted sum; Thus variance is reduced. So RoR (ELU/PELU) is more robust to the inputs than RoR (ReLU).

3.3 RoR Optimization Method

PELU adopts the parameter updating mechanism to make RoR (PELU) more flexible in training. Under conditions of sufficient images in the dataset, ELU in RoR (ELU) has a constant saturation point and exponential decay for negative arguments and a constant slope for positive arguments. The deep RoR (ELU) can more easily get stuck in the saturated part during training where the gradient is almost zero–moreso than RoR (PELU) [18]. The gradient of a certain weight at the layer l in RoR (PELU) containing one neuron in each of its L layers is expressed as (4):

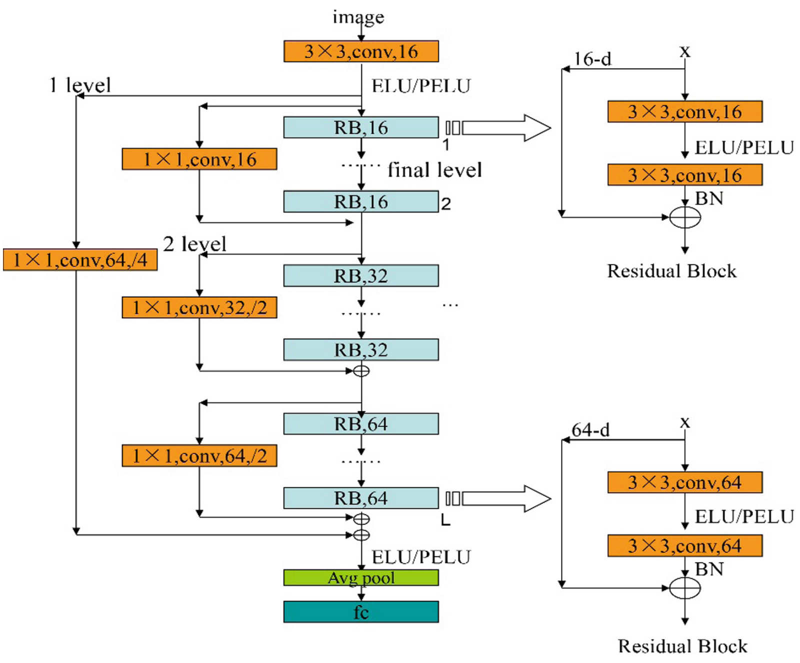

Fig. 1. RoR (ELU/PELU) architecture. The 1-level shortcut is a root-level shortcut, and the remaining three 2-level shortcuts are middle-level shortcuts. The shortcuts in residual blocks are final-level shortcuts.

$$\frac{\partial E}{\partial w_l} = h_{l-1} f'(h_l)[\prod_{j=l+1}^{L} f'(h_j) w_j] \frac{\partial E}{\partial z_L} \tag{4}$$

where h_0 is the input of the network. The output for lst layer is $z_l = f(h_l)$, where $h_l = w_l h_{l-1}$, and loss function is $E = \ell(z_L, y)$ between the network prediction z_L and label y. If $\partial E/\partial w_l$ tends to zero, it will produce vanishing gradients, making the network difficult to converge. One way to overcome this is when:

$f'(h_j)w_j \equiv f'(h_{j-1}w_j)w_j \geq 1$, which means $f'(wh)w \geq 1$. Substituted into the PELU definition, the solution lies in to get:

$$f'(wh)w = \begin{cases} w\frac{a}{b} & if \quad h \geq 0 \\ w\frac{a}{b}\exp(wh/b) & if \quad h < 0 \end{cases} a, b > 0 \tag{5}$$

Vanishing gradients problem is controlled when meeting $f'(wh)w \geq 1$. If $h \geq 0$, $w \geq b/a$ need to be met; $wb/a \cdot \exp(wh/b) \geq 1$ needs to be met, which pushes out $|h| \leq l(w) = |\log(b/aw)|(b/w)$ There is a maximum value of $l(w)$ when $w = \exp(1)b/a$; $l(w) = a\ \exp(-1)$ which means $l(w) \leq a\ \exp(-1)$.

For RoR (ReLU) and RoR (ELU), countering vanishing gradient is mostly possible with positive activations, which causes the bias shift. But for RoR (PELU), a can be adjusted to increase $a\exp(-1)$ and allow more negative activations h to counter vanishing gradients. Moreover, a and b can be adjusted to make which can $w \geq b/a$, which can make RoR (PELU) more flexible in training to eliminate vanishing gradients.

However, in RoR (PELU), additional parameters are added to the activation function compared to RoR (ReLU) and RoR (ELU). Adding the parameter update layer by layer makes the network model more complex. We believe that the parameters in the activation function have a greater impact on performance than parameters of other weight layers. Thus, when the number of images in each category is relatively small, RoR (PELU) is more likely to produce overfitting in training than RoR (ELU). Therefore, ELU will perform better under such conditions. Based on the above analysis, for different datasets of image classification, RoR (PELU) and RoR (ELU) complement each other. So, for different image datasets, we propose an optimization method for RoR based on activation functions:

For RoR, the datasets with more class images (such as CIFAR-10), they should be optimized by PELU, and the RoR (PELU) structure should be adopted. Meanwhile, the datasets with relatively fewer images in each category (such as CIFAR-100) should be optimized by ELU, and the RoR (ELU) structure should be adopted.

3.4 Stochastic Depth and Depth and Width Analysis

Overfitting and vanishing gradients are two challenging issues for RoR, which have a strongly negative impact on performance of image classification. In this paper, to alleviate overfitting, we trained RoR with the drop-path method, and obtained an apparent performance boost. We mitigated the gradient disappearing by appropriately widening the network.

RoR widens the network and adds more training parameters while adding additional shortcuts, which can lead to more serious overfitting problems. Therefore, we used the stochastic depth (SD) algorithm, which is commonly used in residual networks, to alleviate the overfitting problem. We trained our RoR networks by randomly dropping entire residual blocks during training and bypassing their transformations through shortcuts, without performing forward-backward computation or gradient updates. Let p_l mean the probability of the unblocked residual mapping branch of the l th residual block. L is the number of residual blocks, and (6) shows that p_l decreases linearly with the residual block position. p_l indicates that the last residual block is probably unblocked. SD can effectively prevent overfitting problems and reduce training time.

$$p_l = 1 - \frac{l}{L}(1 - p_L) \tag{6}$$

Under the premise network of fixed infrastructure, the main way to improve network performance is to magnify network model by deepening the network. however, increasing the depth of model blindly will lead to worse vanishing gradients. WRN [11] is used to increase width of residual networks to improve the performance, compared to blindly deepened networks (causing the vanishing gradients), in the same order of magnitude, with better performance. Based on this idea, we increased the channel of convolutional layers in the RoR residual blocks from {16, 32, 64} in the original network to $\{16 \times k, 32 \times k, 64 \times k\}$. Feature map dimension extracted from the residual blocks is increased to widen the network, keeping the network from becoming too deep, and further controlling the vanishing gradients problems. A widened RoR network is represented by RoR-WRN.

4 Experiment

In order to analyze the characteristics of three kinds of networks (RoR (ReLU), RoR (ELU), and RoR (PELU)), as well as verify the effectiveness of the optimization scheme, massive experiments were planned. The implementation and results follow.

4.1 Implementation

In this paper, we used RoR for image classification on two image datasets, CIFAR-10 and CIFAR-100. CIFAR-10 is a data set of 60,000 32×32 color images, with 10 classes of natural scene objects containing 6000 images each. Similar to CIFAR-10, CIFAR-100 is a data set of 60,000 32×32 color images, but with 100 classes of natural scene objects. This dataset is just like the CIFAR-10, except it has 100 classes containing 600 images each. The training set and test set contain 50,000 and 10,000 images, respectively. Our implementations were based on Torch 7 with a Titan X. We initialized the weights as in [19]. In both CIFAR-10 and CIFAR-100 experiments, we used SGD with a mini-batch size of 128 for 500 epochs. The learning rate started from 0.1, turned into 0.01 after epoch 250 and to 0.001 after epoch 375. For the SD drop-path method, we set p_l with the linear decay rule of $p_0 = 1$ and $p_L = 0.5$. In RoR-WRN experiments, we set the number of convolution kernels as $\{16 \times k, 32 \times k, 64 \times k\}$ instead of {16, 32, 64} in the original networks. Other architectures and parameters were the same as RoR's. As for the data size being limited in this paper, the experiment adopted two kinds of data expansion techniques: random sampling and horizontal flipping.

4.2 110-Layer RoR Experiments

Three types of 110-layer RoR were used to make up the CIFAR-10/100, with the SD algorithm and without the SD algorithm classification error rate shown in Figs. 2 and 3. It can be seen from the results that RoR+SD can obtain better results than RoR without SD, indicating that SD can effectively alleviate the overfitting problem and improve network performance. Therefore, in the subsequent experiments, we trained RoR with the drop-path method-SD. The results of RoR (ELU) and RoR (PELU) on the CIFAR-

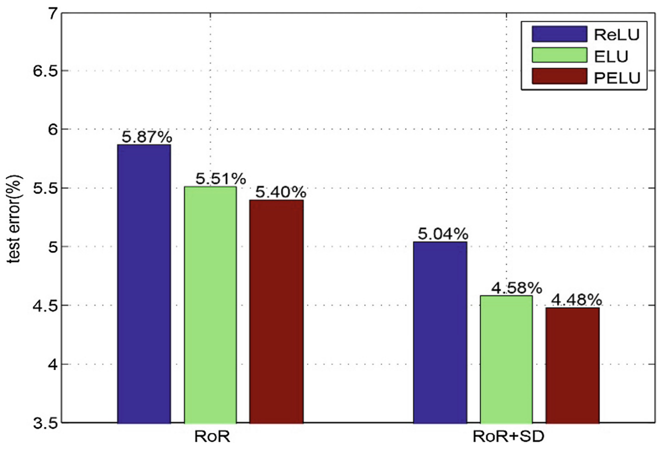

Fig. 2. Test Error (%) on 110-layer RoR

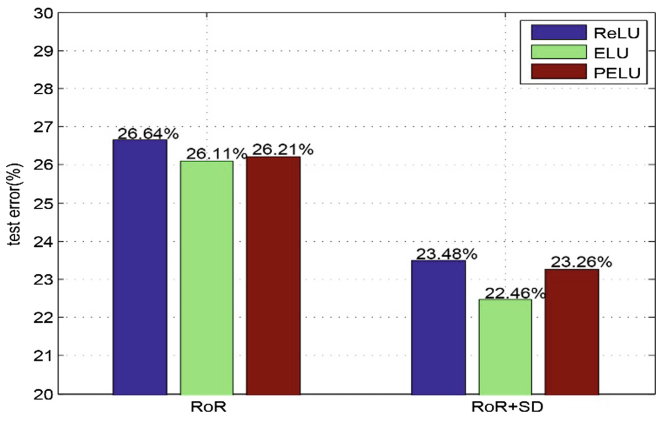

Fig. 3. Test Error (%) on 110-layer RoR

10/100 are better than that of RoR (ReLU). RoR (PELU) obtained lowest test error on CIFAR-10, RoR (ELU) obtained lowest test error on CIFAR-100.

The experimental results perfectly validate the effectiveness of the proposed optimization method. In this paper, we think that in the training, some of the input of ReLU fell into the hard saturation region, resulting in corresponding weight that cannot be updated. In addition, the ReLU output has the offset phenomenon; that is, the output mean value is greater than zero, which will affect the convergence of CNN. Using ELU and PELU, with the left side of the soft saturation, makes it more robust to the inputs, which means, we obtained better results.

4.3 Depth and Width Experiments

In order to further optimize the model, we increased the network model from the two aspects of width and depth. The three types of RoR with 38 layers, k = 2 (RoR-WRN38-2+SD) and 56 layers of network, k = 4 (RoR-WRN56-4+SD) are used for image classification experiments on CIFAR-10/100. The classification test error is shown in Tables 1 and 2. It can be seen from the results that, in the case of widening and deepening appropriately, the three types of RoR have improved performance, while the comparison performance results are basically similar to the 110-layer network. RoR (PELU) on CIFAR-10 obtained the lowest classification test error. As a result, RoR (ELU) obtained the best classification results on CIFAR-100. The results also confirm the optimization method we developed.

Table 1. Test Error (%) on RoR-WRN38-2+SD

	CIFAR-10	CIFAR-100
RoR (ReLU)	4.59	22.48
RoR (PELU)	4.19	22.40
RoR (ELU)	4.28	21.65

Table 2. Test Error (%) on RoR-WRN56-4+SD

	CIFAR-10	CIFAR-100
RoR (ReLU)	3.77	19.73
RoR (PELU)	3.56	20.39
RoR (ELU)	3.98	19.34

4.4 Results Comparison of the Best Model Classification

Table 3 compares our optimized RoR model with the state-of-the-art methods on CIFAR-10/100. It can be seen from Table 3 that the classification test error of RoR-WRN56-4+SD(PELU) and RoR-WRN56-4+SD (ELU) on CIFAR-10/100 is better than that of the original network RoR-WRN56-4+SD(ReLU), which proves the effectiveness of the proposed scheme. It can be seen from the experimental results from the optimized RoR has almost none increase in the computational cost and achieves better classification results compared with the same depth and width network. In view of the good effect of the optimization model, we attempted a deeper RoR-WRN74-4+SD model to obtain the optimal model and achieve state-of-the-art results on CIFAR-10. On the CIFAR-100, RoR-WRN74-4+SD achieves state-of-the-art results in much the same manner according to the amount of parameters as other models. Although ResNeXt and PyramidNet obtained a lower error rate on CIFAR-100, the number of model parameters was much larger than our best model.

Table 3. Test Error (%) on CIFAR-10/100 on different methods

Method(Parameters)	CIFAR-10	CIFAR-100
ELU [17]	6.55	24.28
FractalNet (30 M) [20]	4.59	22.85
ResNets-164 (2.5 M) [8]	5.93	25.16
Pre-ResNets-164 (2.5 M) [9]	5.46	24.33
Pre-ResNets-1001 (10.2 M) [9]	4.62	22.71
ELU-ResNets-110 (1.7 M) [21]	5.62	26.55
PELU-ResNets-110 (1.7 M) [18]	5.37	25.04
ResNets-110+SD (1.7 M) [10]	5.23	24.58
ResNet in ResNet (10.3 M) [24]	5.01	22.90
WResNet-d (19.3 M) [22]	4.70	–
WRN28-10 (36.5 M) [11]	4.17	20.50
CRMN-28 (more than 40 M) [23]	4.65	20.35
RoR-WRN56-4 (13.3 M) [13]	3.77	19.73
multi-resnet (145 M) [25]	3.73	19.60
DenseNet (27.2 M) [14]	3.74	19.25
PyramidNet (28.3 M) [27]	3.77	18.29
ResNeXt-29, 16 $\times$ 64d (68.1 M) [26]	3.58	17.31
RoR-WRN56-4+SD(PELU)(13.3 M)	**3.56**	–
RoR-WRN56-4+SD(ELU)(13.3 M)	–	**19.34**
RoR-WRN74-4+SD(PELU)(18.2 M)	**3.52**	–
RoR-WRN74-4+SD(ELU)(18.2 M)	–	**19.07**

5 Conclusion

In this paper, we put forward an optimization method of Residual Networks of Residual Networks (RoR) by analyzing performance of three activation functions. We acquired amazing image classification results on CIFAR-10 and CIFAR-100. The experiment results show optimizational RoR can give more control over bias shift and vanishing gradients and get excellent image classification performance.

Acknowledgement. This work is supported by National Power Grid Corp Headquarters Science and Technology Project under Grant No. 5455HJ170002(Video and Image Processing Based on Artificial Intelligence and its Application in Inspection), National Natural Science Foundation of China under Grants No. 61302163, No. 61302105, No. 61401154 and No. 61501185.

References

1. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
2. Hong, C., Yu, J., Wan, J., et al.: Multimodal deep autoencoder for human pose recovery. IEEE Trans. Image Process. **24**(12), 5659–5670 (2015)
3. Hong, C., Yu, J., Tao, D., et al.: Image-based three-dimensional human pose recovery by multiview locality-sensitive sparse retrieval. IEEE Trans. Industr. Electron. **62**(6), 3742–3751 (2015)

4. Krizhenvshky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional networks. In: Proceedings of the Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
5. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge, arXiv preprint arXiv:1409.0575 (2014)
6. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014)
7. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Van-houcke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision Pattern Recognition, pp. 1–9 (2015)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition, arXiv preprint arXiv:1512.03385 (2015)
9. He, K., Zhang, X., Ren, S., Sun, J.: Identity mapping in deep residual networks, arXiv preprint arXiv:1603.05027 (2016)
10. Huang, G., Sun, Y., Liu, Z., Weinberger, K.: Deep networks with stochastic depth, arXiv preprint arXiv:1605.09382 (2016)
11. Zagoruyko, S., Komodakis, N.: Wide residual networks, arXiv preprint arXiv:1605.07146 (2016)
12. Krizhenvshky, A., Hinton, G.: Learning multiple layers of features from tiny images. M.Sc. thesis, Dept. of Comput. Sci., Univ. of Toronto, Toronto, ON, Canada (2009)
13. Zhang, K., Sun, M., Han, X., et al.: Residual networks of residual networks: multilevel residual networks. IEEE Trans. Circuits Syst. Video Technol. **PP**(99), 1 (2016)
14. Huang, G., Liu, Z., Weinberger, K.Q., et al.: Densely connected convolutional networks, arXiv preprint arXiv:1608.06993 (2016)
15. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Networks **5**(2), 157–166 (2014)
16. Nair, V., Hinton, G.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the ICML, pp. 807–814 (2010)
17. Clevert, D.-A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus), arXiv preprint arXiv:1511.07289 (2015)
18. Trottier, L., Giguere, P., Chaibdraa, B.: Parametric exponential linear unit for deep convolutional neural networks, arXiv preprint arXiv:1605.09322 (2016)
19. Mishkin, D., Matas, J.: All you need is a good init, arXiv preprint arXiv:1511.06422 (2015)
20. Larsson, G., Maire, M., Shakhnarovich, G.: FractalNet: ultra-deep neural networks without residuals, arXiv preprint arXiv:1605.07648 (2016)
21. Shah, A., Shinde, S., Kadam, E., Shah, H.: Deep residual networks with exponential linear unit, arXiv preprint arXiv:1604.04112 (2016)
22. Shen, F., Zeng, G.: Weighted residuals for very deep networks, arXiv preprint arXiv:1605.08831 (2016)
23. Moniz, J., Pal, C.: Convolutional residual memory networks, arXiv preprint arXiv:1606.05262 (2016)
24. Targ, S., Almeida, D., Lyman, K.: Resnet in resnet: generalizing residual architectures, arXiv preprint arXiv:1603.08029 (2016)
25. Abdi, M., Nahavandi, S., Multi-residual networks, arXiv preprint arXiv:1609.05672 (2016)
26. Xie, S., Girshick, R., Dollr, P., et al.: Aggregated residual transformations for deep neural networks, arXiv preprint arXiv:1611.05431 (2016)
27. Han, D., Kim, J., Kim, J.: Deep pyramidal residual networks, arXiv preprint arXiv:1610.02915 (2016)