A Lightweight Encoder-Decoder Path for Deep Residual Networks

Xin Jin, Yanping Xie, Xiu-Shen Wei, *Member, IEEE,* Bo-Rui Zhao, Yongshun Zhang, Xiaoyang Tan, and Yang Yu

***Abstract*—In this paper, we present a novel lightweight path for deep residual neural networks. The proposed method integrates a simple plug-and-play module, *i.e.*, a convolutional Encoder-Decoder (ED), as an augmented path to the original residual building block. Thanks to the abstract design and ability of the encoding stage, the decoder part tends to generate feature maps where highly semantically relevant responses are activated while irrelevant responses are restrained. By a simple element-wise addition operation, the learned representations derived from the identity shortcut and original transformation branch are enhanced by our ED path. Furthermore, we exploit lightweight counterparts by removing a portion of channels in the original transformation branch. Fortunately, our lightweight processing does not cause an obvious performance drop, but bring computational economy. By conducting comprehensive experiments on ImageNet, MS-COCO, CUB200-2011 and CIFAR, we demonstrate the consistent accuracy gain obtained by our ED path for various residual architectures, with comparable or even lower model complexity. Concretely, it decreases the top-1 error of ResNet-50 and ResNet-101 by 1.22% and 0.91% on the task of ImageNet classification, and increases the mmAP of Faster R-CNN with ResNet-101 by 2.5% on the MS-COCO object detection task. Code is available at https://github.com/Megvii-Nanjing/ED-Net.**

***Index Terms*—Convolutional neural networks, deep learning, encoder-decoder, residual networks, base model.**

I. Introduction

DEEP Convolutional Neural Networks (DCNNs) have achieved a series of breakthroughs in various fundamental computer vision tasks, such as image classification [26], [16], object detection [12], [35], [51], semantic segmentation [29], and many other tasks [24], [7], [46], [1], [9], [33], [45]. These successes mainly derive from the discriminative representation learned by DCNNs. The deep representation can also generalize well to many other different tasks after supervised training on large scale image datasets *e.g.*, ImageNet [37] and MS-COCO [28].

• X. Jin and Y. Xie contributed equally to this work. X.-S. Wei (with Nanjing University of Science and Technology) is the corresponding author.
• X. Jin and B.-R. Zhao are with Megvii Research Nanjing, Megvii Technology, Nanjing, China. X.-S. Wei is with PCA Lab, Key Lab of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, and Jiangsu Key Lab of Image and Video Understanding for Social Security, School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. Y. Zhang is with with the National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China. Y. Xie and X. Tan are with College of Computer Science and Technology, Nanjing University of Aeroncautics and Astronautics, MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China. Y. Yu is with Polixir Technologies Ltd., Nanjing, China.
• Email: {jinxin, zhaoborui}@megvii.com, weixs.gm@gmail.com, {nuaaxyp, x.tan}@nuaa.edu.cn, zhangys@lamda.nju.edu.cn, yuy@nju.edu.cn.

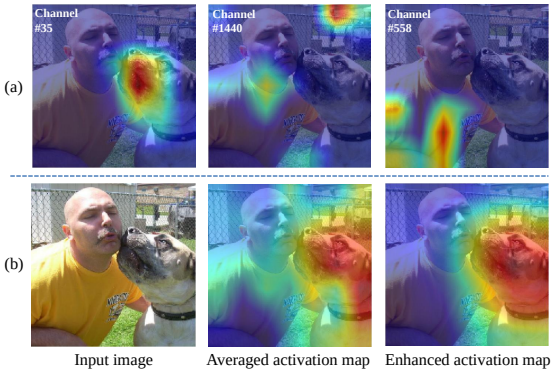


Figure 1: Visualization of the learnt concept of "dog" under cluttered background by using the original residual network *vs*. our ED path augmented one. **First row** (a) shows three activation maps from the input channels of the last building block of ResNet-50. **Second row** (b) presents the input image, activation map averaged over all input channels, and our enhanced activation map – it properly highlights the discriminative regions and meanwhile restrains irrelevant backgrounds.

Convolutional filters are the core of DCNNs. DCNNs are constructed by stacking a series of convolutional layers, combined with non-linear activation functions and down-samplings. By end-to-end joint training, these filters are optimized according to the final loss function, and generate distributed representations of input data in high-level layers [4]. However, as shown in the first row of Figure 1, for a test image labeled "dog", it is common to observe that some irrelevant backgrounds still have highly activated responses. Although these noisy activations might be suppressed by the following layers, in theory, they still have a chance to hurt the recognition accuracy.

Aiming at the aforementioned issues, recent works attempt to improve the representational ability of DCNNs by refining intermediate feature responses with attention mechanisms [43], [19], [47]. For example, Hu *et al.* [19] introduced Squeeze-and-Excitation (SE) blocks to explicitly capture the inter-dependencies between channel-wise feature responses. SE blocks have demonstrated good accuracy gains over various deep architectures, including ResNet [16] and ResNeXt [48] series. Woo *et al.* [47] proposed Convolutional Block Attention Module (CBAM) which augments existing SE blocks with an additional spatial attention module. However, the spatial attention of CBAM is highly dependent on SE blocks, and cannot perform well individually. In this paper, we propose a simple module, which can be integrated into any deep residual

architecture without dependence on other attention modules. Our method attends on object of interests more accurately than previous methods (cf. Figure 4), and achieves better recognition accuracy on various vision tasks, *e.g.*, general image classification, object detection, instance segmentation and fine-grained classification.

Our motivation relies on two key insights. First, instead of learning augmented attention weights, a specialized module can be designed to directly extract the most informative features from redundant inputs, and then enhance the CNN representations with the extracted features. Secondly, since the popular encoder-decoder architecture can efficiently extract compact and informative codes through purely unsupervised learning [41], it can be expected to extract discriminative features from redundant inputs through supervised training.

We elaborate the effectiveness of our method under the deep residual learning framework [16], [48]. Specifically, we keep the original structure of residual building blocks unchanged as they are crucial for training very deep architectures, but augment them with a parallel convolutional encoder-decoder (ED) path (see Figure 2) to enhance the learned representations. Driven by the dimensionality reduction of encoders, the decoder part tends to generate feature maps where highly semantically relevant responses are activated while the irrelevant responses are restrained. By a simple element-wise addition operation, the learned representations derived by identity shortcuts and the original transformation branch can be enhanced by these attended decoder responses. Furthermore, compared with the activation units of input features, the activation units in the decoder features have larger receptive fields, and thus can incorporate more useful context information into the learned representations. This property is beneficial for context-dependent down-stream recognition tasks, such as object detection and instance segmentation.

For efficiency, we equip our ED path with grouped convolutions as a default option to reduce model and computational complexity. Furthermore, we exploit a lightweight processing strategy for residual networks augmented with our ED path. By removing a portion of channels in the original transformation branch, we find that it brings lower floating point operations (FLOPs), but does not cause obvious accuracy drops.

We integrate our ED path into several residual architectures, including the ResNet [16] and ResNeXt [48] series, and also the Squeeze-and-Excitation (SE) networks [19]. We provide extensive evaluations and analyses on ImageNet [37], CUB-200-2011 [42], and CIFAR [25] for image classification, and on MS-COCO [28] for object detection and instance segmentation. With our ED path, we achieve consistent accuracy gains for various residual architectures, *e.g.*, decreasing the top-1 error by 1.22% and 0.91% for ResNet-50 and ResNet-101 on ImageNet classification, and for MS-COCO object detection, increasing mmAP by 1.9% and 2.5% for Faster R-CNN [35] with ResNet-50 and ResNet-101 as backbones. Furthermore, our ED path can be conveniently combined with state-of-the-art deep architectures (*e.g.*, SENets [19]) to further improve recognition accuracy. In addition, our lightweight ED counterpart of ResNet-50, which almost reduces half of floating point operations, but still outperforms ResNet-50 in accuracy on ImageNet classification.

To the best of our knowledge, our method is the first attempt to exploit the ability of supervised discriminative feature extraction of encoder-decoder structures, for enhancing the representational power of deep residual architectures. The main contributions of this paper are as follows:

- We propose a simple but effective encoder-decoder path for deep residual learning, which can enhance the learned representations through an element-wise addition operation.
- We explore a lightweight design for deep residual networks attached with our encoder-decoder path, which does not cause obvious performance drops, but brings computational economy.
- We conduct extensive experiments to verify the effectiveness of our ED path. Both quantitative and qualitative results validate that our proposal can benefit various deep residual architectures on diverse vision tasks.

The rest of the paper is organized as follows. Section II retrospects the related works. Section III details proposed method and Section IV describes the implementation details. Experiments and analysis are provided in the Section V, followed by the conclusion part in Section VI.

II. Related work

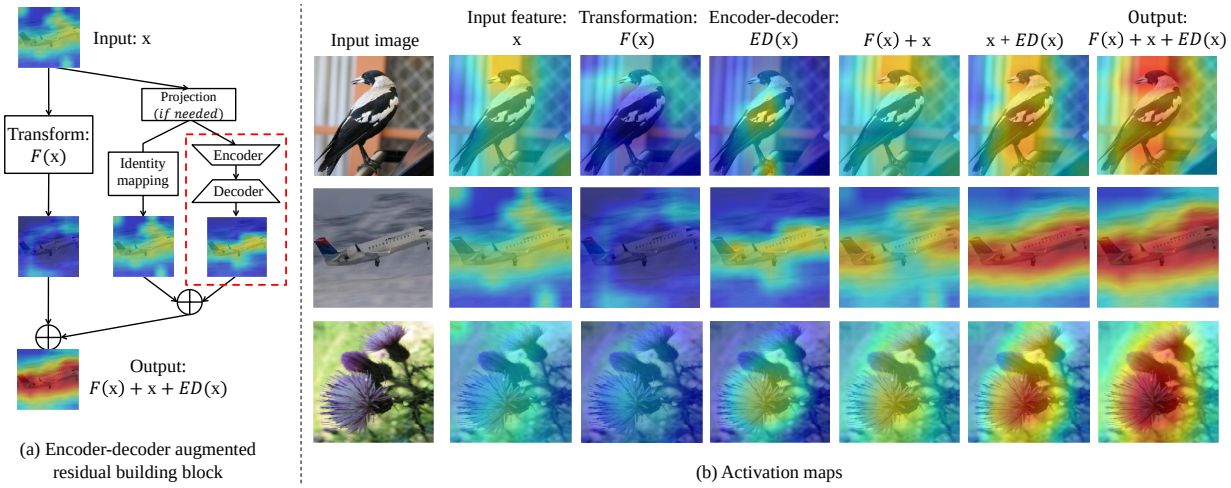
In this section, we briefly review some closely related works, including deep residual networks, encoder-decoder architectures, and group convolutions.

A. Deep residual networks

Deep convolutional neural networks were significantly improved by the residual learning paradigm [16] introduced in 2016. Residual networks [16] use identity mappings as shortcut connections [5], [36], letting the transformation layer to fit a residual mapping, rather than the original unreferenced one. This modification simplifies the optimization of very deep networks, leading to record-breaking performance on challenging vision tasks, such as ImageNet classification and MS-COCO object detection.

Very recently, researches show that extending the transformation layers of ResNets into multi-branch structures can improve the generalization ability [48], [38]. For example, ResNeXt [48] uses a set of transformations as residual units, and shows that increasing the size of the set of transformations is able to improve classification accuracy. Previous multi-branch residual units mainly feature efficient model design under the restricted condition of maintaining complexity. Instead, our goal of augmenting residual building blocks with an ED path is to extract the discriminative information from redundant inputs, and refine the learned representations by a simple element addition operation.

Invertible ResNets (i-ResNets) [3] is recently proposed to define a generative model that can be trained by maximum likelihood on unlabeled data, with a tractable approximation to the Jacobian log-determinant of a residual block. i-ResNets provide a unified model paradigm for classification, density estimation, and generation. Instead of equipping the classification CNNs



(a) Encoder-decoder augmented residual building block

(b) Activation maps

Figure 2: **Left:** a residual building block augmented with our encoder-decoder path. **Right:** example images illustrating the activations from different paths in `conv5_3` block of ResNet-50. The input feature maps seem to have rather even activations, and some background may also be activated. In contrast, the feature maps produced by our encoder-decoder module tend to generate focused activations which embody in the most discriminative regions of the image.

with the ability of generative modeling, our encoder-decoder structures aim to extract the discriminative information from layer inputs, which can benefit the representational power and improve the classification accuracy of the residual network family.

It is worth noting that our convolutional encoder-decoder structures significantly differ from conventional bottleneck structures, although performing channel reduction and restoration. Firstly, instead of using 1×1 layer for dimension reduction and restoration, we directly apply 3×3 convolutional/deconvolutional layer as encoder and decoder, where the convolutional layer performs dimension reduction and the deconvolutional layer performs dimension restoration. Secondly, our encoder-decoder structures first down-sample the feature maps and then recover them with a scaling step of 2, while all the convolutional layers in conventional bottleneck residual blocks produce output maps of the same size in the same network stage. Essentially, these differences are due to the fact that our encoder-decoder structures aim to extract the most informative features from layer inputs by the means of spatial dimensionality reduction, while the bottleneck residual blocks aims to reduce the computational cost when increasing the depth of network.

B. Encoder-decoder architectures

The encoder-decoder architectures have been widely used in dimensionality reduction [17], feature learning [20], [23], [10], and semantic segmentation [32], [2]. For instance, using stacked restricted Boltzmann machines as the encoder [17], the extracted low-dimensional representations can recover the structure of input data, much better than linear projection methods like principal component analysis. Convolutional encoder, when used for unsupervised feature learning, can learn powerful representations that generalize better in high-level vision tasks than hand-crafted features [20]. In recent years, convolutional encoder-decoder architectures are widely adapted to solve various challenging vision tasks [32], [2], [31]. In this paper, we validate another perspective of convolutional encoder-decoder that it is beneficial to extract the most informative features in an image when used as an augmented path of a CNN model.

Previous works, which integrate encode-decoder modules into CNNs, mainly aim to equip CNNs with the ability of generative modeling or unsupervised learning. Instead, our goal is to enhance deep residual architectures with encode-decoder modules to improve the representational power. Due to the distinct difference in motivation, our method does not involve a reconstruction loss adopted by previous works, letting the encoder-decoder path only to extract discriminative representations from redundant layer inputs. In [50], a generative path (trained by reconstruction) is integrated into a discriminative convolutional architecture, yielding good accuracy on a variety of semi-supervised and supervised tasks. In [34], a novel reconstruction-based framework is designed for effective identity and non-identity feature disentanglement. These two works differ from ours in both motivation and implementation, and lack the ability of improving the classification accuracy of the very deep ResNet family.

C. Grouped convolutions

Group convolution can be formally defined as: A grouped convolutional layer separates the input channels into groups, and then apply separated convolutional filters for each group, and concatenate the output map of each group independently to form the final output map. If the number of groups is equal to the number of channels, then this layer performs channel-wise convolution.

Grouped convolution is firstly introduced in AlexNet [26] (if not earlier) for deploying the network on two GPUs, each processing one group of feature maps. Ever since, group convolutions have been widely adopted in CNN architecture design, both for large networks like Inception series [39], [40], [38]

and ResNeXt [48] pursuing high accuracy, and for lightweight models like Xception [8], MobileNet [18] and ShuffleNet [49] customized for mobile devices with very limited computing power. In this paper, we use group convolutions to reduce the computational and parameter complexity of our encoder-decoder branch. In particular, while ResNeXt can be regarded as a multi-branch variant of ResNet, our method differs with ResNeXt in the fact that we keep the transformation branches of ResNet and ResNeXt unchanged, and augment them with a novel encoder-decoder path to further enhance the learned representations. In contrast, ResNeXt directly replaces the original transformation branches of ResNet with the grouped multi-branch structures.

III. Approach

In this section, we elaborate our design choices of the encoder-decoder (ED) path for residual networks, and then show visualization and qualitative analyses, and finally present discussions with some closely related works.

A. Encoder-decoder paths for ResNets

1) Grouped encoder-decoder module: Our goal is to design an effective encoder-decoder module, which can be conveniently applied with the building blocks of state-of-the-art deep residual learning architectures. Among many possible choices, we opt for the simplest. Concretely, we use a 3×3 convolutional operation with stride of 2 as the encoder, and use the same filter size and stride for the decoder (a transpose convolution layer). We follow the design principle of the transformation branch in ResNets [16], and experimentally set the number of encoder filters to be the number of 3×3 filters used by the transformation branch, and the number of decoder filters to be the dimension of output channels of the block.

Directly equipping residual building blocks with augmented encoder-decoder module will certainly cause considerable computational burdens. To address this drawback, we employ grouped convolutions to strike a good trade-off between model performance and complexity. Specifically, we split the 3×3 convolution filters in all encoders and decoders into G groups ($G = 32$ in defaults), reducing the model and computational complexity to a large extent. It is worth noting that our accuracy gain is derived from the encoder-decoder path rather than grouped convolutions, which will be verified by experiments in Section V-D.

2) Integration into ResNet and ResNeXt: It is convenient and straightforward to integrate our encoder-decoder module into any deep residual architectures. In this work, we mainly focus on ResNet [16] and ResNeXt [48] series. Figure 2 (a) shows a general encoder-decoder augmented residual building block, and Figure 3 demonstrates the building blocks of our ED-ResNet-50 and ED-ResNeXt-50. We note that our architecture differs from ResNeXt, which improves ResNet by replacing the original transformation branch with a multi-branch structure. Instead, we keep the original architectures of both ResNet and ResNeXt unchanged, but augment them with a novel encoder-decoder path. Detailed architecture comparisons of our ED-ResNet-50 with the original ResNet-50 are presented in Table I.

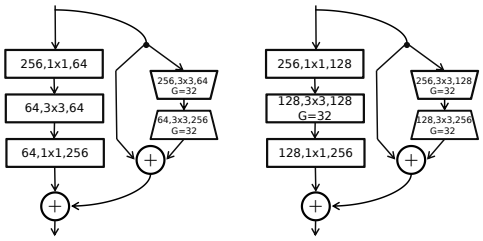


Figure 3: Building block of ED-ResNet-50 / ED-ResNeXt-50.

We introduce the encoder-decoder module into residual building blocks as an augmented branch, and summarize the outputs from the identity shortcut, original transformation path and the encoder-decoder path as the final output of the building block. We demonstrate that this simple element-wise addition operation is effective to utilize the informative features carried by the encoder-decoder path to refine learned representations. Coincidently, similar enhancement strategy based on element-wise addition is also employed in [27] to combine multi-level features for object detection.

It is worth noting that when changing the dimension of input/output channels across stages, we perform a shared linear projection for both identity shortcuts and encoder-decoder paths to match the dimensions. This design originates from the fact that the projection shortcuts, which have shown to be very effective for increasing dimensions [16], can be regarded as an approximation of original inputs. Thus, it can be directly fed into our encoder-decoder path.

For notation simplicity, in the following, we term ResNets and ResNeXts with our augmented encoder-decoder paths as "**ED-ResNets**" and "**ED-ResNeXts**". Both ED-ResNets and ED-ResNeXts are easy to implement by current open-source deep learning toolboxes.

3) Lightweight implementations: Equipping original residual networks with our ED paths will cause a slight increment on the number of model parameters and FLOPs, as shown in Table I (25.5M $\rightarrow$ 27.7M for the number of parameters and $4.1 \times 10^9 \rightarrow 4.3 \times 10^9$ for FLOPs). For computational efficiency, and also for fair comparisons, we explore lightweight implements of ED-ResNets and ED-ResNeXts by cutting a portion of channels in the original transformation branch.

The intuition is that since the encoder-decoder modules can extract discriminative representations from redundant inputs, they should to some extent reduce the burden of representation learning for original transformation branches. We empirically observe that even when removing half of 3×3 convolutional filters in original transformation branches[1], ResNet-50 with our ED path still outperform the original network, while the FLOPs is reduced to 51%. More detailed quantitative results about the tradeoff between model accuracy and computational complexity are given in Table VI.

[1]After the channel pruning, the number of 1×1 filters in previous layers are also required to adjust accordingly.

Table I: **Left:** Original ResNet-50. **Right:** Our ED-ResNet-50 with 3×3 convolutional encoder (stride = 2) and 3×3 convolutional decoder equipped with grouped convolutions ($\sharp$ group = 32).

Stage	Output	Repeat	**ResNet-50**		**Our ED-ResNet-50**		
conv_1	112×112	1	7×7, 64, stride=2		7×7, 64, stride=2		
max-pooling	56×56	1	3×3, 64, stride=2		3×3, 64, stride=2		
–	–	–	Transformation	Shortcuts	Transformation[1]	Encoder-decoder	Shortcuts
conv_2	56×56	3	$[1 \times 1, 64]$ $[3 \times 3, 64]$ $[1 \times 1, 256]$	Identity mapping	$[1 \times 1, 64]$ $[3 \times 3, 64]$ $[1 \times 1, 256]$	[conv, 3×3, 64] group=32 [deconv, 3×3, 256] group=32	Identity mapping
conv_3	28×28	4	$[1 \times 1, 128]$ $[3 \times 3, 128]$ $[1 \times 1, 512]$	Identity mapping	$[1 \times 1, 128]$ $[3 \times 3, 128]$ $[1 \times 1, 512]$	[conv, 3×3, 128] group=32 [deconv, 3×3, 512] group=32	Identity mapping
conv_4	14×14	6	$[1 \times 1, 256]$ $[3 \times 3, 256]$ $[1 \times 1, 1024]$	Identity mapping	$[1 \times 1, 256]$ $[3 \times 3, 256]$ $[1 \times 1, 1024]$	[conv, 3×3, 256] group=32 [deconv, 3×3, 1024] group=32	Identity mapping
conv_5	7×7	3	$[1 \times 1, 512]$ $[3 \times 3, 512]$ $[1 \times 1, 2048]$	Identity mapping	$[1 \times 1, 512]$ $[3 \times 3, 512]$ $[1 \times 1, 2048]$	[conv, 3×3, 512] group=32 [deconv, 3×3, 2048] group=32	Identity mapping
cls	1×1	1	Global average-pooling 1000-d fc,softmax		Global average-pooling 1000-d fc,softmax		
$\sharp$ parameters			25.5M		27.7M [2]		
FLOPs			4.1×10^9		4.3×10^9		

[1]Here, the "transformation" means the original transformation branch. Strictly, both the original transformation branch and our encoder-decoder branch play the role of learning the transformation of input features.
[2]The number of parameters and FLOPs of our original ED-ResNet-50 can be further reduced by our lightweight implementations, *e.g.*, cutting channels in the original transformation branches. Empirical studies validate that cutting channels can produce a comparable model complexity, but our model can achieve better classification accuracy, cf. Table VI.

B. Visualization and analyses

To intuitively interpret the role played by our encoder-decoder paths, we visualize the activation maps from different branches (as well as the combinations of them) in a residual building block. The individual branches include the original transformation branch, the identity shortcut branch and our encoder-decoder branch. We follow the process in [44] for feature visualization. The activation tensor produced by the last convolutional block of ResNets or ResNeXts is firstly obtained, and then added up through the depth direction to get a 2-D matrix. The 2-D matrix is then resized to the size of input image, and visualized by OpenCV API to demonstrate the activations on the input image.

Figure 2 gives the visualization results. In each row, the combined activation maps are computed by directly adding up individual maps. We can see that the input feature response $\mathbf{x}$ might have a wide range of spatial activations, including some irrelevant backgrounds. Taking $\mathbf{x}$ as inputs, the encoder-decoder branch $\mathcal{ED}(\mathbf{x})$ is able to produce focused activations on the most discriminative regions of input images, while suppressing irrelevant backgrounds. By direct element addition operation, the encoder-decoder branch $\mathcal{ED}(\mathbf{x})$ can effectively enhance original input features, and thus generate better final image representations.

We also note that the output of augmented building block is $\mathcal{F}(\mathbf{x})+\mathbf{x}+\mathcal{ED}(\mathbf{x})$, where $\mathcal{F}(\mathbf{x})$ and $\mathcal{ED}(\mathbf{x})$ can be viewed as two different transformation branches. During joint end-to-end training, these two branches seem to interact with each other. However, as shown in Figure 2 (b), we empirically find that the original transformation branch $\mathcal{F}(\mathbf{x})$ (in the 3^{rd} column) and the encoder-decoder branch $\mathcal{ED}(\mathbf{x})$ (in the 4^{th} column) play quite different roles in high-level residual building blocks. $\mathcal{F}(\mathbf{x})$ tends to learn a very small perturbation w.r.t. the input $\mathbf{x}$, while $\mathcal{ED}(\mathbf{x})$ tends to extract more informative features from $\mathbf{x}$. The output of $\mathcal{F}(\mathbf{x})$ is consistent with the observation in [16] – high-level convolutional layers in ResNets produce very small magnitudes of responses. This, from another side, suggests that our encoder-decoder paths do not impose obvious impact on the function of original transformation branches.

C. Comparisons to SENets and CBAM

We would like to emphasize that, our method is significantly different from traditional attention based methods like SENets [19] and CBAM [47]. Firstly, we do not learn additional attention weights to refine feature responses. Instead, we utilize an encoder-decoder module to directly extract the most informative features from raw inputs, and then refine the learned representation by the simple element-wise addition. Secondly, the abstract design of the ED path allows us to reduce the computational burden of the original transformation branch in ResNets without obvious accuracy drops, which however has not been discovered in literature by attention based methods. Thirdly, our method does not depend on any attention modules. While, CBAM is an augmented contribution highly dependent on SENets, as it inserts additional spatial attention into SENets.

Since our method improves the representational power of CNNs from a novel perspective, it can be conveniently combined with the attention based methods to further improve recognition accuracy. As validated in experiments, our ED path, when integrated with SENets, consistently improves the baselines for ImageNet classification, and outperforms CBAM in many cases (cf. Table IV). Furthermore, although CBAM further exploits spatial attention on the basis of SENets, our method still obtains consistently better results than CBAM on challenging tasks including object detection (cf. Table X and Table XII), instance segmentation (cf. Table XI) and fine-grained recognition (cf. Table XIII).

IV. Implementation Details

Following the practices in [26], [16] on the ImageNet dataset, the input images are 224×224 patches with the per-pixel

Table II: Comparisons to **baselines** on **ImageNet-1K**.

Model	Top-1 err.	Top-5 err.
ResNet-50 [16]	24.34	7.32
ResNet-50 + 2conv	23.75	6.98
ResNet-50 + ED (Ours)	**23.12**	**6.54**
ResNet-101 [16]	23.12	6.52
ResNet-101 + 2conv	22.69	6.34
ResNet-101 + ED (Ours)	**22.21**	**6.23**
ResNet-152 [16]	22.44	6.37
ResNet-152 + 2conv	22.39	6.34
ResNet-152 + ED (Ours)	**21.98**	**6.09**
ResNeXt-50 [48]	22.59	6.41
ResNeXt-50 + 2conv	22.56	6.37
ResNeXt-50 + ED (Ours)	**22.01**	**6.11**
ResNeXt-101 [48]	21.34	5.66
ResNeXt-101 + 2conv	21.30	5.63
ResNeXt-101 + ED (Ours)	**20.93**	**5.32**

mean subtracted, randomly cropped from resized images with standard data augmentation and random horizontal flipping. Optimization is performed by stochastic gradient descent with momentum 0.9 and a mini-batch size of 256 on 8 GPUs, and the weight decay is 0.0001. We start from a learning rate of 0.1, and divide it by 10 every 30 epochs, all models are trained for 100 epochs. We adopt the weight initialization proposed in [15]. When testing, we apply a single crop evaluation on the validation set, where 224×224 pixels are center cropped from each image whose shorter edge is first resized to 256. We perform batch normalization [22] after the convolutions, then ReLU [30] is applied as the non-linear activation function. Notably, in the ED path, ReLU is not performed after the batch normalization in convolutional decoder to avoid removing negative responses. On the CIFAR-10 dataset [25], 4 pixels are first padded on each side of the image during training, then we randomly crop 32×32 pixels from the padded image or its horizontal flip as inputs. In testing, we keep the original image size unchanged. All models are trained for 300 epochs and the learning rate is 0.1 which is divided it by 10 in the schedule of 150 and 225. We use a weight decay of 0.0001, a momentum of 0.9, and a mini-batch size of 128 on 2 GPUs.

V. Experiments

We conduct four series of experiments to verify the effectiveness of the proposed encoder-decoder (ED) path for deep residual networks, including ImageNet-1K [37] classification, MS-COCO [28] object detection and instance segmentation, fine-grained image recognition on CUB200-2011 [42], and ablation studies on CIFAR-10 [25]. We re-implement all the models by PyTorch for fair comparisons.

A. ImageNet classification

On ImageNet-1K [37] classification, we first compare our method to baselines and SENets, then validate the effectiveness of our ED path under lower computational cost.

1) Comparisons to baselines: We consider ResNet and ResNeXt series as the most important baselines of our method. Despite our efforts to reproduce the reported results, there are many factors to effect the re-implemented accuracies, to name a few, deep learning frameworks, undisclosed training tricks, etc. In fact, our reimplemented ResNet series perform better than the reported ones in the SENet paper, but our re-implemented ResNeXt series are slightly inferior. We also implement a simple baseline by replacing the encoder-decoder path with a two-layer 3×3 (grouped) convolution process that has the same model complexity. This aims to justify the advantages of ED path over the augmented vanilla convolutional path.

Table III: Comparisons to **ResNets** for **ImageNet-1K** classification on top-1/5 errors, parameters, FLOPs and latency.

Model	Top-1 err.	Top-5 err.	parameters	FLOPs	latency
ResNet-50	24.34	7.32	25.5×10^6	4.1×10^6	0.18s
ResNet-50 + ED (Ours)	**23.12**	**6.54**	26.7×10^6	4.3×10^9	0.19s
ResNet-101	23.12	6.52	49.1×10^6	7.9×10^9	0.38s
ResNet-101 + ED (Ours)	**22.21**	**6.23**	51.4×10^6	8.2×10^9	0.40s
ResNet-152	22.44	6.37	72.8×10^6	11.7×10^9	0.55s
ResNet-152 + ED (Ours)	**21.98**	**6.09**	76.3×10^6	12.2×10^9	0.58s

Table IV: Improvements over **SENets** and **CBAM** on **ImageNet** classification.

Model	Top-1 err.	Top-5 err.
SE-ResNet-50 [19]	23.27	6.59
SE-ResNet-50 + CBAM [47]	22.66	6.31
SE-ResNet-50 + ED (Ours)	**22.46**	**6.28**
SE-ResNet-101 [19]	22.37	6.13
SE-ResNet-101 + CBAM [47]	**21.51**	**5.69**
SE-ResNet-101 + ED (Ours)	21.71	6.04
SE-ResNeXt-50 [19]	21.61	5.72
SE-ResNeXt-50 + CBAM [47]	21.92	5.91
SE-ResNeXt-50 + ED (Ours)	**21.20**	**5.64**
SE-ResNeXt-101 [19]	21.32	5.54
SE-ResNeXt-101 + CBAM [47]	21.07	5.59
SE-ResNeXt-101 + ED (Ours)	**20.89**	**5.30**

As shown in Table II, ResNets and ResNeXts with ED path consistently outperform the baseline architectures. In particular, it decreases the top-1 error of ResNet-50 and ResNet-101 by 1.22% and 0.91%, respectively. ED-ResNet-50 has a top-1 error of 23.12%, similar to much deeper architecture ResNet-101 which almost has double FLOPs. Similarly, the top-1 and top-5 errors of ED-ResNet-101 is 22.21% and 6.23%, outperforming deeper ResNet-152. Our ED variants of ResNets and ResNeXts are consistently superior to "two-conv" baselines, although the latter can also improve original ResNets and ResNeXts. We note that the augmented "two-conv" path, when equipped with ResNeXts, could only obtain very minimal improvements compared to our ED path. It conjectures that this is because the grouped "two-conv" path has similar behaviors with the original transformation branch of ResNeXt, and thus can hardly introduce more useful information during training.

Besides accuracy, we further conducted extensive experiments to show the comparisons between our methods with ED path and the baseline ResNet series on top-1/5 errors, parameters, FLOPs, and latency. We measure the latency with batch size 1 on single score of Intel Xeon CPU E5-2690. As shown in Table III, our ED module adds minimal additional cost on parameters, FLOPs, and latency. We argue that the added cost is negligible, considering the accuracy gains brought by the ED path. For example, our ED-ResNet-50 takes 0.19s to process an image and ResNet-101 takes 0.38s, while the top-1/5 errors of our ED-ResNet-50 are similar to ResNet-101. In the future, we will investigate deeply about how to design lightweight CNN architectures with our ED path.

2) Improvements over State-of-the-Art: Our method improves residual networks from a different perspective with re-

Table V: Improvements over **Res2Nets** on **ImageNet-1K**.

Model	Top-1 err.	Top-5 err.
Res2Net-50	22.01	6.15
Res2Net-50 + ED (Ours)	**21.21**	**5.56**
Res2NeXt-50	21.76	6.09
Res2NeXt-50 + ED (Ours)	**21.03**	**5.42**
SE-Res2Net-50	21.56	5.94
SE-Res2Net-50 + ED (Ours)	**21.02**	**5.31**
Res2Net-101	20.81	5.57
Res2Net-101 + ED (Ours)	**20.01**	**5.08**

cent state-of-the-art CNN architectures, including SENets [19], CBAM [47], and Res2Net [11]. Therefore, we integrate our ED path into these architectures to investigate whether it can further boost the recognition accuracy. As shown in Table IV, our ED version of SENets achieve reasonable improvements over original SENets. ED path can further decrease the top-1 error of SE-ResNet-50 and SE-ResNeXt-50 by 0.81% and 0.41%. Particularly, SE-ResNeXt-50 with our ED path even outperforms much deeper SE-ResNeXt-101. For CBAM [47], it improves SENets by combining global average- and max-pooling features, and combines an additional spatial attention module with SE blocks. In Table IV, we also compare our method with CBAM. Our ED variants of SE-ResNet-50 and SE-ResNet-101 achieves comparable performance with CBAM, while the ED variants of SE-ResNeXt-50 and SE-ResNeXt-101 outperforms the corresponding CBAM counterparts.

We also integrate our encoder-decoder path into recent state-of-the art Res2Net [11] series that enhance the original ResNet series with multi-scale representation. Table V gives the comparison results with the Res2Net series. The encoder-decoder version of Res2Nets consistently outperform original Res2Nets for various network architectures, including ResNet, ResNeXt and SE-Net. In particular, even for the deep Res2Net-101 architecture, our encoder-decoder path can still reduce the top-1 and top-5 error rate on ImageNet-1K classification by 0.8% and 0.49%, respectively. These results demonstrate that our encoder-decoder path can improve deep residual networks in conjunction with multi-scale representations.

3) Comparisons under comparable FLOPs: The proposed encoder-decoder path imposes a slight increment in model and computational burden to the baseline networks. To perform an apple-to-apple comparison, we remove a portion of channels from original transformation branches of ED-Nets to align their FLOPs with baseline networks. Specifically, we directly remove $\{4, 8, 16, 32\}$ channels for 3×3 transform convolutional layers from conv_2 to conv_5 stages in ED-ResNet series; while for ED-ResNeXt series, we remove $\{20, 40, 80, 160\}$ channels from 3×3 convolutional layers at different stages, and then split the channels in all stages into 36 groups. We denote these reduced architectures as "**ED-ResNet-A**" / "**ED-ResNeXt-A**".

As shown in Table VI, both ED-ResNet-A and ED-ResNeXt-A consistently outperform ResNets and ResNeXts, in the same setting, by a significant margin. For example, ED-ResNet-50-A decreases the top-1 and top-5 errors of the baseline ResNet-50 by 1.26% and 0.85%, even slightly better than the original ED-ResNet-50 without channel pruning. ED-ResNet-101-A decreases the top-1 error of ResNet-101 by 0.89%, which is comparable with the performance of original ED-ResNet-101. These results suggest that our encoder-decoder paths can, to some extent, reduce the computational overhead of the transformation branches, due to their enhancement over the learned representations. In addition, since our ED-ResNeXt-A changes the original group of ResNeXt from 32 to 36, we also implement a modified version of ResNeXt-50 for fair comparisons. This new architecture is ResNeXt-50-36×3d, which has 36 groups while the width of bottleneck is 3 (the same as ED-ResNeXt-50-A). ResNeXt-50-36×3d achieves top-1 error of 23.04% and top-5 error of 6.46%, which is obviously inferior to our ED-ResNeXt-50-A.

Table VI: Results of **efficient ED-Nets** for **ImageNet-1K classification**, by removing a portion of 3×3 convolutional filters from original transformation branches.

Model	top-1 err.	top-5 err.	FLOPs
ResNet-50 [16]	24.34	7.32	4.1×10^9
ED-ResNet-50-A[1]	**23.08**	**6.47**	4.0×10^9
ED-ResNet-50-B[2]	23.94	6.95	$\mathbf{2.1\times10^9}$
ResNet-101 [16]	23.12	6.52	7.9×10^9
ED-ResNet-101-A	**22.23**	**6.24**	7.8×10^9
ED-ResNet-101-B	23.14	6.49	$\mathbf{3.9\times10^9}$
ResNet-152 [16]	22.44	6.37	11.7×10^9
ED-ResNet-152-A	**22.01**	**6.11**	11.5×10^9
ED-ResNet-152-B	22.52	6.41	$\mathbf{5.6\times10^9}$
ResNeXt-50 [48]	22.59	6.41	4.2×10^9
ED-ResNeXt-50-A	**22.03**	**6.12**	4.2×10^9
ED-ResNeXt-50-B	22.61	6.43	$\mathbf{2.9\times10^9}$
ResNeXt-101 [48]	21.34	5.66	8.0×10^9
ED-ResNeXt-101-A	**20.97**	**5.33**	7.9×10^9
ED-ResNeXt-101-B	21.57	5.71	$\mathbf{5.4\times10^9}$

[1]"-A" means that we remove a portion of channels of the transform branches to make ED-Net have the same/comparable FLOPs as that of the baseline network.
[2]"-B" means that we remove **half** of the 3×3 convolution filters of the transform branches to make ED-Net have much less FLOPs than that of the baseline network.

4) Exploring efficient ED-Nets: We ulteriorly exploit very efficient implementations of ED-Nets by removing half channels of 3×3 convolutional layers at all stages for ED-ResNet and ED-ResNeXt series. This greatly reduces FLOPs of original ED-Nets, *i.e.*, reducing 49% and 31% FLOPs of the original ResNet-50 and ResNeXt-50, respectively. We denote these efficient architectures as "**ED-ResNet-B**" / "**ED-ResNeXt-B**".

As reported in Table VI, although we greatly reduce FLOPs, ED-ResNet-B and ED-ResNeXt-B still obtains comparable or even slightly better performance than the baseline networks. For example, ED-ResNet-50-B still achieves better performance than the baseline ResNet-50, and ED-ResNeXt-50-B only suffers from very slight performance degradation compared to ResNeXt-50. These results suggest that: 1) the encoder-decoder modules allow us to lighten the computational burden of original transformation branches to a large extent, without obvious performance degradation; 2) encoder-decoder modules have the potential in the model compression and efficient model design, which is a promising future work direction.

In addition, to verify whether the efficient design property is unique to the ED path, we take ED-ResNet-50-B for example, and replace the ED path with a two-layer 3×3 (grouped) convolution process. The top-1 and top-5 error of this two-conv alternative are 24.42% and 7.59%, respectively, which are inferior to both the ResNet-50 and our ED-ResNet-50-B.

5) Ablation studies: We conduct ablation studies on ImageNet-1k to investigate three questions. Firstly, can we

Table VII: Comparisons between original ResNets (ResNeXts), our ED-Nets, and ED counterparts **without** identity mapping on ImageNet-1k classification, where "w." means "with", "w/o." means "without", and "ID map" means "identity mapping".

Model	Top-1 err.	Top-5 err.
ResNet-50	24.34	7.32
ResNet-50 w. ED (Ours)	**23.12**	**6.54**
ResNet-50 w. ED w/o. ID map	25.19	7.73
ResNet-101	23.12	6.52
ResNet-101 w. ED (Ours)	**22.21**	**6.23**
ResNet-101 w. ED w/o. ID map	32.96	12.41
ResNet-152	22.44	6.37
ResNet-152 w. ED (Ours)	**21.98**	**6.09**
ResNet-152 w. ED w/o. ID map	42.66	19.38
ResNeXt-50	22.59	6.41
ResNeXt-50 w. ED (Ours)	**22.01**	**6.11**
ResNeXt-50 w. ED w/o. ID map	23.99	7.14
ResNeXt-101	21.34	5.66
ResNeXt-101 w. ED (Ours)	**20.93**	**5.32**
ResNeXt-101 w. ED w/o. ID map	32.97	12.67

Table VIII: Comparisons between original ResNets (ResNeXts), our ED-Nets, and the counterparts by replacing all transformations with our ED modules on ImageNet-1k classification, where "w." means "with" and "w/o." means "without".

Model	Top-1 err.	Top-5 err.
ResNet-50	24.34	7.32
ResNet-50 w. ED (Ours)	**23.12**	**6.54**
ResNet-50 w. ED w/o. transform	31.07	11.37
ResNet-101	23.12	6.52
ResNet-101 w. ED	**22.21**	**6.23**
ResNet-101 w. ED w/o. transform	30.01	10.77
ResNet-152	22.44	6.37
ResNet-152 w. ED	**21.98**	**6.09**
ResNet-152 w. ED w/o. transform	29.03	10.10
ResNeXt-50	22.59	6.41
ResNeXt-50 w. ED	**22.01**	**6.11**
ResNeXt-50 w. ED w/o. transform	29.13	10.12
ResNeXt-101	21.34	5.66
ResNeXt-101 w. ED	**20.93**	**5.32**
ResNeXt-101 w. ED w/o. transform	28.22	9.49

discarding the identity mapping of ResNets (ResNeXts) when having our ED paths? Secondly, can we replacing all the transformation branches of ResNets (ResNeXts) by our ED branches? Thirdly, can we benefit from pre-trained ResNets (ResNeXts) when attaching our ED paths?

For the first question, we conduct a series of experiments by discarding identity mapping of deep residual architectures while having our ED path. As shown in Table VII, for CNNs with moderate depth (*e.g.*, ResNet-50 and ResNeXt-50), replacing the identity mapping with our ED path leads to a slight accuracy drop. While for very deep residual networks such as ResNet-101 and ResNet-152, discarding the identity mapping while having our ED path leads to significant performance degradation. These results demonstrate that the identity shortcuts are crucial for solving the degradation problem when training very deep neural networks (*e.g.*, ResNet-101 and ResNet-152), and can be hardly replaced by other structures. However, when training networks with moderate depth (*e.g.*, ResNet-50), our encoder-decoder paths can to some extent play the role of identity shortcuts, as they can convey the discriminative features from layer inputs.

For the second question, we have conduct a series of experiments by replacing all transformation branches with our ED modules. Table VIII shows the comparison results. For all deep residual architectures, this modification leads to significant performance degradation. We believe that there are two key factors responsible for these results. Firstly, our encoder-decoder structures are designed as a feature enhancement component of ResNet family, which themselves are not sufficient for the feature extraction task by the original transformation branches. Secondly, our encoder-decoder structures have much less (about 1/20) computational complexity (FLOPs), compared to the transformation branches of ResNets and ResNeXts. Although our encoder-decoder modules can not directly replace all transformation branches, with our ED modules, we can greatly reduce the number of filters of the original transformation branches without causing an obvious performance drop.

Table IX: Comparisons between original ResNets (ResNeXts), our ED-Nets, and ED counterparts with **pre-trained** ResNets (ResNeXts) on ImageNet-1k classification, where "w." means "with".

Model	Top-1 err.	Top-5 err.
ResNet-50	24.34	7.32
ResNet-50 w. ED	23.12	**6.54**
ResNet-50 (pre-trained) w. ED	**22.71**	6.60
ResNet-101	23.12	6.52
ResNet-101 w. ED	**22.21**	**6.23**
ResNet-101 (pre-trained) w. ED	22.62	6.41
ResNet-152	22.44	6.37
ResNet-152 w. ED	**21.98**	**6.09**
ResNet-152 (pre-trained) w. ED	22.55	6.28
ResNeXt-50	22.59	6.41
ResNeXt-50 w. ED	**22.01**	**6.11**
ResNeXt-50 (pre-trained) w. ED	22.44	6.19
ResNeXt-101	21.34	5.66
ResNeXt-101 w. ED	**20.93**	**5.32**
ResNeXt-101 (pre-trained) w. ED	21.87	6.10

For the third question, we conduct a series of experiments that fine-tune the pre-trained ResNets and ResNeXts, leaving our encoder-decoder modules randomly initialized. We multiply the original learning rate on ImageNet by 0.1 to prevent the model from overfitting. Table IX shows the comparison results. Interestingly, compared to the **original** ResNets and ResNeXts, the pre-training process (along with our randomly initialized encoder-decoder structures) can benefit the residual architectures with moderate depth (*e.g.*, 50 layers), but leads to negligible performance improvement or even performance degradation for deeper networks such as ResNet-152 and ResNeXt-101. When compared with our **encoder-decoder** counterparts of ResNets and ResNeXts, the pre-training process does not show advantages, and causes performance degradation in most cases. We speculate that the improvement by pre-training for ResNets with moderate depth can be owed to better initialization, as well as our encoder-decoder structures, while the degradation problem for very deep networks might be caused by overfitting. We consider in-deep analyses about these results as our future work.

B. MS-COCO experiments

We further evaluate the generalization ability of our encoder-decoder modules on object detection and instance segmentation by using the MS-COCO dataset [28], which contains 80k training images and 40k validation images.

1) Object detection: We train Faster R-CNN [35] as our detection test bed, and then evaluate it on the 40k validation

Table X: **Object detection** results on **MS-COCO** 40k validation set using **Faster R-CNN** [35].

Model	mmAP	AP@0.50	AP@0.75	AR100
ResNet-50 [16]	31.0	50.9	33.1	44.0
ResNet-50 + CBAM [47]	31.6	51.8	33.3	45.4
ResNet-50 + ED (Ours)	**32.9**	**53.0**	**35.4**	**45.8**
ResNet-101 [16]	32.5	52.0	34.9	45.4
ResNet-101 + CBAM [47]	34.2	54.4	36.1	46.2
ResNet-101 + ED (Ours)	**35.0**	**54.9**	**37.5**	**47.9**

Table XI: **Instance segmentation** results on **MS-COCO** 40k validation set using **Mask R-CNN** [14].

Model	mmAP	AP@0.50	AP@0.75	AR100
ResNet-50 [16]	29.2	50.0	30.3	40.7
ResNet-50 + CBAM [47]	28.4	48.6	29.4	40.7
ResNet-50 + ED (ours)	**30.3**	**51.4**	**31.6**	**41.5**
ResNet-101 [16]	30.6	52.0	31.9	42.0
ResNet-101 + CBAM [47]	30.3	52.0	31.7	42.5
ResNet-101 + ED (ours)	**31.6**	**53.2**	**33.0**	**42.5**

Table XII: **Object detection** results on **MS-COCO** 40k validation set using **Mask R-CNN** [14].

Model	mmAP	AP@0.50	AP@0.75	AR100
ResNet-50 [16]	34.1	54.0	36.8	46.8
ResNet-50 + CBAM [47]	32.9	52.4	35.1	46.5
ResNet-50 + ED (Ours)	**35.5**	**55.1**	**38.1**	**47.9**
ResNet-101 [16]	36.3	56.1	39.0	49.0
ResNet-50 + CBAM [47]	36.0	56.1	38.5	49.4
ResNet-101 + ED (Ours)	**37.5**	**57.3**	**40.1**	**49.6**

Table XIII: **Fine-grained recognition** results on **CUB200-2011** [42].

Model	Err. (depth=50)	Err. (depth=101)
ResNet [16]	15.14	14.51
ResNet + CBAM [47]	15.01	14.40
ResNet + ED (Ours)	**14.84**	**14.35**
ResNeXt [48]	14.52	14.23
ResNeXt + CBAM [47]	14.41	14.11
ResNeXt + ED (Ours)	**13.91**	**13.53**

images. The training code is based on [6], and we keep the default settings for fair comparisons.

As shown in Table X, when using as the backbone, our ED-ResNets can significantly boost the performance of Faster R-CNN for general object detection. In particular, ED-ResNet-50 outperforms ResNet-50 by 1.9% on MS-COCO standard metric mmAP, and improves AP@IoU=0.75 by 2.3%, which is even more significant than the improvement at AP@IoU=0.50. It conjectures that this is because our ED paths help the network to produce more accurate activations for objects of interests, which can greatly ease the training of the regression branch of Faster R-CNN, and lead to more accurate localization results. Furthermore, ED-ResNet-101 improves 2.5% mmAP for ResNet-101 based Faster R-CNN, which is a significant improvement in object detection.

Also, there might be another important reason accounting for our superior performance in object detection – the enlarged receptive field, which is crucial for detecting some context-dependent objects. Specifically, the spatial down-sampling operation of the encoding stage enlarges the receptive fields of the activation units of feature maps, and the decoding stage keeps the receptive fields of the feature map units unchanged. Then, through the element-wise addition, our encoder-decoder path naturally incorporates more context information into the learned representations. In the furture, the impacts of our encoder-decoder path on the task of object detection can be deeply investigated.

2) Instance segmentation: Instance segmentation is a challenging task as it requires correct prediction of pixel-level object masks. We train Mask R-CNN [14] as the instance segmentation test bed. Mask R-CNN is a general framework that can predict both bounding boxes and pixel-level masks for object of interests. We follow the codes of [13], and re-implement it with PyTorch.

Table XI and Table XII show the instance segmentation and object detection results by Mask R-CNN, respectively. ResNets with encoder-decoder paths consistently improve the performance of baseline ResNet for both tasks. For example, ED-ResNet-50 outperforms ResNet-50 by 1.1% mmAP for instance segmentation, and by 1.4% mmAP for object detection. Furthermore, our ED-Nets show consistent advantages over CBAM [47] when used as backbone for both object detection and instance segmentation.

C. Fine-grained image recognition

We further evaluate our method for fine-grained visual recognition using the CUB200-2011 dataset [42] that contains 11,788 images of 200 bird species. Besides, CUB200-2011 also provides accurate instance mask annotations for birds, which allows us to conduct a quantitative investigation about the attention (highlighted) regions automatically generated by the deep models. For a test image, the attention region is expected to be the object of interests. For this dataset, we resize all images to 448×448 in our experiments.

1) Classification accuracy: We compared our method with ResNets and ResNeXts baselines, as well as the SENets and CBAM version of them. As shown in Table XIII, our method consistently achieves the best results under different depths, showing its potential for fine-grained visual recognition.

2) Quantitative investigation about attention regions: We perform quantitative evaluations on CUB200-2011 to validate that our ED-based model can attend more accurate region of the main object of interest. We follow [44] for localizing objects (*i.e.*, obtaining the attention regions). Specifically, the activation tensor produced by the last convolutional block of ResNets and ResNeXts is firstly obtained, and then added up through the depth direction to get a 2-D matrix. In the following, we compute the mean value of the matrix and regard it as the threshold for localizing the object. We use the Intersection over Union (IoU) between the attention region and the ground truth mask as the metric of the "attention accuracy".

The attention region (object localization) is shown in Figure 4, which clearly shows that the ED path is beneficial to remove the redundant activations and/or extract the most informative features. The quantitative results are shown in Table XIV, where our method consistently outperforms the baseline methods.

We also investigate the attention regions produced by the input features and output features of our ED path. As shown in Table XV, the attention regions produced by the decoder

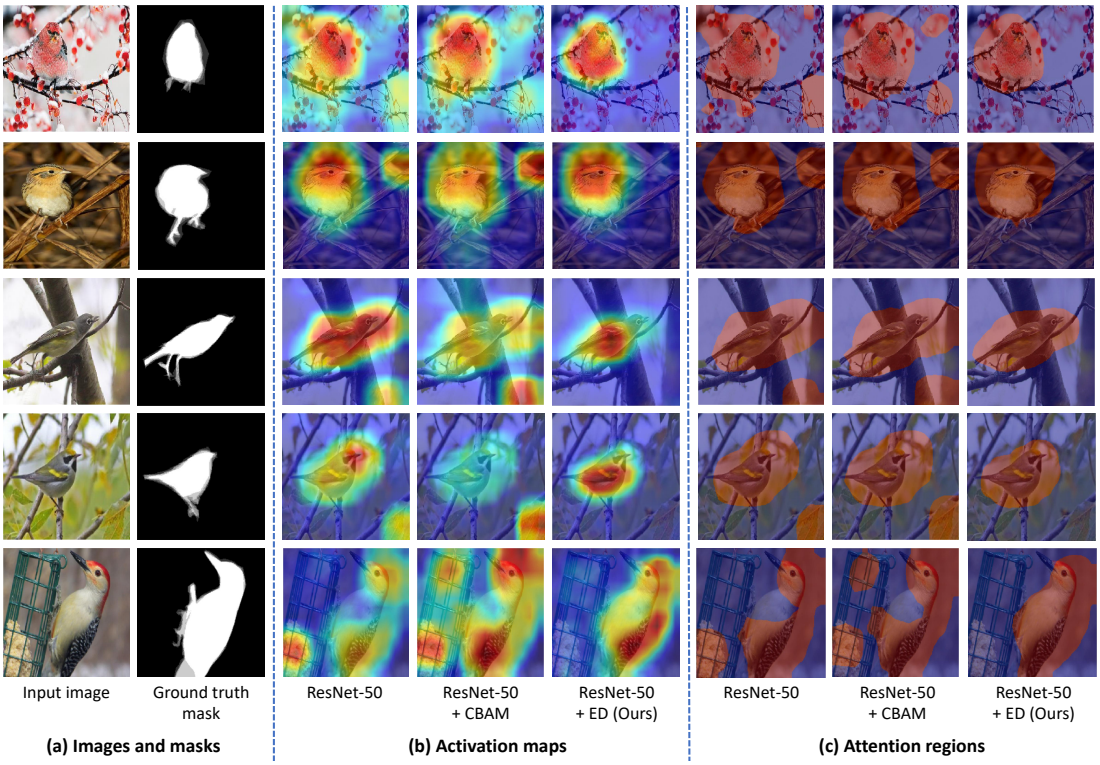


Figure 4: Visualization of attention regions for some images from CUB200-2011, calculated for the last convolutional outputs by different models. **(a) Images and masks**, where five bird images and corresponding ground truth masks are shown. **(b) Activation maps** produced by ResNet-50, CBAM-integrated ResNet-50 (ResNet-50 + CBAM), and our ED-integrated ResNet-50 (ResNet-50 + ED) respectively. **(c) Attention regions** by thresholding the raw activation maps with the mean value, which follows [44].

Table XIV: **Quantitative comparisons of attention regions** generated by our methods and baselines on **CUB200-2011** [42].

Model	IoU (depth=50)	IoU (depth=101)
ResNet [16]	54.99	55.57
ResNet + CBAM [47]	58.93	59.04
ResNet + ED (Ours)	**59.86**	**59.97**
ResNeXt [48]	55.03	55.62
ResNeXt + CBAM [47]	58.98	59.18
ResNeXt + ED (Ours)	**59.92**	**60.12**

Table XV: **Quantitative comparisons of attention regions** generated by the **input** and **decoder** features of our ED path on **CUB200-2011** [42].

Feature	IoU (depth=50)	IoU (depth=101)
ResNet ED Input	51.76	52.43
ResNet ED Output	**53.92**	**54.51**
ResNeXt ED Input	52.08	52.97
ResNeXt ED Output	**54.17**	**55.49**

features of our ED path are more accurate than those produced by the input features, which clearly shows that our ED path is beneficial to extract the most informative features and/or remove the redundant activations. Note that since our ED path is only one path of the building block of ED-integrated ResNet and ResNeXt, the "attention accuracy" of the decoder feature is inferior to that of the full output of the building block, which summarizes the output from the original transformation branch, identity shortcut and our ED path.

Table XVI: Comparisons between our **ED** proposal and a **two-layer convolution** for **ResNet** on **CIFAR-10**.

ResNet	Baseline	Our ED	2Conv
20	7.79	**7.34**	7.59
32	7.22	**6.65**	6.80
44	6.99	**6.20**	6.34
56	6.44	**5.95**	6.29
110	5.77	**5.67**	6.13

Table XVII: Comparisons between our **ED** proposal and a **two-layer convolution** for **ResNeXt-29** on **CIFAR-10**.

ResNeXt-29	Baseline	Our ED	2Conv
Full transform	3.62	**3.59**	3.78
25% channels of 3×3 conv	–	**3.41**	3.86
50% channels of 3×3 conv	–	**3.56**	3.75
75% channels of 3×3 conv	–	**3.50**	3.84
Wider ResNet	4.17	–	–

D. More Ablation studies on CIFAR-10

1) Encoder-decoder vs. Two-layer convolutions: We compare our method with the two-layer convolution baselines on CIFAR-10 [25] to further justify the effectiveness of our encoder-decoder proposal. The two-conv baselines are implemented by simply replacing the encoder-decoder path with a two-layer 3×3 convolution process which has the same model complexity.

For ResNet series, as reported in Table XVI, both encoder-decoder paths and two-layer convolution paths can boost the

Table XVIII: Trade-off between **accuracy** and **complexity** with and without **grouped convolutions** (♯ group = 16).

ResNet	Baseline		ED w/o group		ED with group	
	error	♯ param	error	♯ param	error	♯ param
20	7.79	0.27M	**6.78**	0.56M	7.34	**0.30M**
32	7.22	0.46M	**6.48**	0.94M	6.65	**0.51M**
44	6.99	0.66M	**5.99**	1.34M	6.20	**0.72M**
56	6.44	0.85M	**5.78**	1.73M	5.95	**0.92M**
110	5.77	1.70M	**5.51**	3.40M	5.67	**1.84M**

performance of ResNets with different depth, but encoder-decoder consistently performs better. For ResNeXt, we adopt the ResNeXt-29 architecture used in [48] for fair comparison, and vary the ratio of kept channels of 3×3 original transformation convolutional layers for more in-depth investigation. We observe that the most lightweight version of ED-ResNeXt-29 achieves the best performance, while the two-layer convolution shortcuts are inferior to the baseline ResNeXt-29 (cf. Table XVII). These results also confirm the efficiency of our encode-decoder architectures.

2) Impacts of grouped convolutions: Recent studies [21] show that grouped convolutions can greatly reduce model and computational complexity without obvious performance drop in accuracy. We further investigate whether it happens for our encoder-decoder proposal.

As shown in Table XVIII, both encoder-decoder counterparts with and without grouped convolutions outperform the baseline architectures. We observe that the grouped convolutions cause a slight drop in accuracy compared to the ED version without grouped convolutions, which justifies that our accuracy improvement mainly derives from the encoder-decoder structure rather than grouped convolutions. On the other hand, since grouped convolutions significantly reduce the computational burden of encoder-decoder modules, we integrate them into all encoder-decoder paths.

3) Dimensionality of encoder feature maps: We investigate the effects of the reduced spatial dimensionality of ED on the final classification performance. By changing the padding numbers on the inputs and the stride size (fixing the convolution kernel as 3×3), we can get different reduction ratios of the input feature map at the middle layer of ED. The results are shown in Figure 5. It is clear to see that, when the reduction ratio equals 50%, it achieves the best performance (the padding number is 1, and the stride size is 2). Besides, we can also observe that different reduction ratios will not affect the final classification accuracy hardly. The fluctuation scope is less than 0.35%, which demonstrates the robustness of our method.

4) Quantitative analyses of decoder activations: We perform quantitative analyses about the responses of original input features and decoder features. We threshold the deep responses by 0 and compute the percentage of activated responses in the feature map cells from the last block of 56-layer ED-ResNet. Figure 6 shows the results obtained by 10,000 test images from CIFAR. It clearly shows that the number of activated responses becomes lower after our ED proposal (45.97%→20.37%), which is a quantitative explanation about ED can produce focused responses compared to input feature maps. Besides, as illustrated in Figure 2, these focused responses tend to highlight the discriminative regions of the inputs.

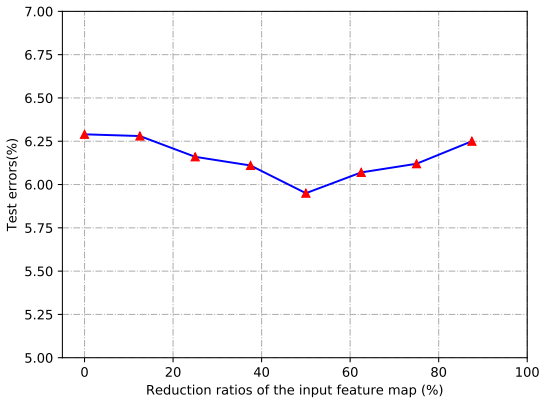


Figure 5: Impact about the reduction ratios by ED.

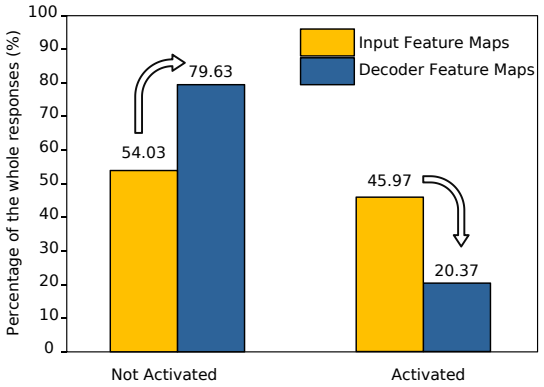


Figure 6: Distribution of the activated deep responses from input feature and decoder feature maps.

VI. Conclusion

In this paper, we proposed a novel lightweight residual learning path for deep neural networks, implemented by a convolutional encoder-decoder module. It can be regarded as an augmented path parallel with the existing identity shortcuts and the original transformation branch. Thanks to the abstract design and ability of the encoding stage, the decoder part tends to both highlight the highly semantically relevant deep activations and restrain the irrelevant or noisy deep responses. By removing a portion of channels in the original transformation branch, we can obtain a lightweight version of ED-Nets, without causing obvious accuracy drop. Extensive experiments on several large-scale datasets validated that our proposal consistently benefits various residual architectures on large-scale image classification, object detection, instance segmentation and fine-grained recognition.

Acknowledge

The authors would like to thank the editor and the anonymous reviewers for their critical and constructive comments and suggestions. This work is partially supported by National Science Foundation of China (61976115, 61672280, 61732006). Y. Xie and Y. Zhang's contribution was made when they were interns in Megvii Research Nanjing.

REFERENCES

[1] A. Aimar, H. Mostafa, E. Calabrese, A. Rios-Navarro, R. Tapiador-Morales, I.-A. Lungu, M. B. Milde, F. Corradi, A. Linares-Barranco, S.-C. Liu, et al. Nullhop: A flexible convolutional neural network accelerator based on sparse representations of feature maps. *IEEE Trans. Neural Netw. & Learn. Syst.*, 30(3):644–656, 2018. 1

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(12):2481–2495, 2017. 3

[3] J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, and J.-H. Jacobsen. Invertible residual networks. In *Proc. Int. Conf. Mach. Learn.*, pages 573–582, 2019. 2

[4] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, 2013. 1

[5] C. Bishop, C. M. Bishop, et al. *Neural networks for pattern recognition*. Oxford university press, 1995. 2

[6] X. Chen and A. Gupta. An implementation of Faster RCNN with study for region sampling. *arXiv preprint arXiv:1702.02138*, 2017. 9

[7] J. Cheng, J. Wu, C. Leng, Y. Wang, and Q. Hu. Quantized CNN: a unified approach to accelerate and compress convolutional networks. *IEEE Trans. Neural Netw. & Learn. Syst.*, 29(10):4730–4743, 2017. 1

[8] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1251–1258, 2017. 3

[9] R. J. Cintra, S. Duffner, C. Garcia, and A. Leite. Low-complexity approximate convolutional neural networks. *IEEE Trans. Neural Netw. & Learn. Syst.*, 29(12):5981–5992, 2018. 1

[10] A. Creswell and A. A. Bharath. Denoising adversarial autoencoders. *IEEE Trans. Neural Netw. & Learn. Syst.*, 30(4):968–984, 2018. 3

[11] S. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. H. Torr. Res2net: A new multi-scale backbone architecture. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019. 7

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 580–587, 2014. 1

[13] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. https://github.com/facebookresearch/detectron, 2018. 9

[14] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2980–2988, 2017. 9

[15] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1026–1034, 2015. 6

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 770–778, 2016. 1, 2, 4, 5, 6, 7, 9, 10

[17] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 3

[18] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 3

[19] J. Hu, L. Shen, and G. Sun. Squeeze-and-Excitation networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 7132–7141, 2018. 1, 2, 5, 6, 7

[20] F. J. Huang, Y.-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1–8, 2007. 3

[21] Y. Ioannou, D. Robertson, R. Cipolla, A. Criminisi, et al. Deep roots: Improving CNN efficiency with hierarchical filter groups. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1231–1240, 2017. 11

[22] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. Int. Conf. Mach. Learn.*, pages 448–456, 2015. 6

[23] M. Kachuee, S. Darabi, B. Moatamed, and M. Sarrafzadeh. Dynamic feature acquisition using denoising autoencoders. *IEEE Trans. Neural Netw. & Learn. Syst.*, 2018. 3

[24] J. Kim, A.-D. Nguyen, and S. Lee. Deep CNN-based blind image quality predictor. *IEEE Trans. Neural Netw. & Learn. Syst.*, 30(1):11–24, 2018. 1

[25] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 2, 6, 10

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proc. Advances in Neural Inf. Process. Syst.*, pages 1097–1105, 2012. 1, 3, 5

[27] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2117–2125, 2017. 4

[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *Proc. Eur. Conf. Comp. Vis.*, pages 740–755, 2014. 1, 2, 6, 8

[29] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3431–3440, 2015. 1

[30] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. Int. Conf. Mach. Learn.*, pages 807–814, 2010. 6

[31] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *Proc. Eur. Conf. Comp. Vis.*, pages 483–499, 2016. 3

[32] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1520–1528, 2015. 3

[33] N. Passalis and A. Tefas. Training lightweight deep convolutional neural networks using bag-of-features pooling. *IEEE Trans. Neural Netw. & Learn. Syst.*, 30(6):1705–1715, 2018. 1

[34] X. Peng, X. Yu, K. Sohn, D. N. Metaxas, and M. Chandraker. Reconstruction-based disentanglement for pose-invariant face recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1623–1632, 2017. 3

[35] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Proc. Advances in Neural Inf. Process. Syst.*, pages 91–99, 2015. 1, 2, 8, 9

[36] B. D. Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007. 2

[37] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. ImageNet large scale visual recognition challenge. *Int. J. Comput. Vision*, 115(3):211–252, 2015. 1, 2, 6

[38] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, Inception-ResNet and the impact of residual connections on learning. In *Proc. Conf. AAAI*, pages 4278–4284, 2017. 2, 3

[39] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1–9, 2015. 3

[40] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the Inception architecture for computer vision. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2818–2826, 2016. 3

[41] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, 2010. 2

[42] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 2, 6, 9, 10

[43] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang. Residual attention network for image classification. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 3156–3164, 2017. 1

[44] X.-S. Wei, J.-H. Luo, J. Wu, and Z.-H. Zhou. Selective convolutional descriptor aggregation for fine-grained image retrieval. *IEEE Trans. Image Process.*, 26(6):2868–2881, 2017. 5, 9, 10

[45] X.-S. Wei, P. Wang, L. Liu, C. Shen, and J. Wu. Piecewise classifier mappings: Learning fine-grained learners for novel categories with few examples. *IEEE Trans. Image Process.*, 28(12):6116–6125, 2019. 1

[46] X.-S. Wei, C.-W. Xie, J. Wu, and C. Shen. Mask-CNN: Localizing parts and selecting descriptors for fine-grained bird species categorization. *Pattern Recogn.*, 76:704–714, 2018. 1

[47] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon. CBAM: Convolutional block attention module. In *Proc. Eur. Conf. Comp. Vis.*, pages 1–14, 2018. 1, 5, 6, 7, 9, 10

[48] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 5987–5995, 2017. 1, 2, 3, 4, 6, 7, 9, 10, 11

[49] X. Zhang, X. Zhou, M. Li, and J. Sun. ShuffleNet: an extremely efficient convolutional neural network for mobile devices. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 6848–6856, 2018. 3

[50] J. Zhao, M. Mathieu, R. Goroshin, and Y. Lecun. Stacked what-where auto-encoders. *arXiv preprint arXiv:1506.02351*, 2015. 3

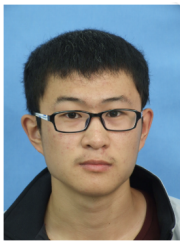
[51] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu. Object detection with deep learning: A review. *IEEE Trans. Neural Netw. & Learn. Syst.*, 2019. 1

Xin Jin received the BS, MS, and PhD degrees from the Department of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2009, 2012, and 2017, respectively. He is currently a researcher with Megvii Research Nanjing. His research interests include computer vision and deep learning, especially focusing on face landmark detection and general object detection.

Yanping Xie received the BS degree from the Department of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2017, and is currently working toward the MS degree. His research interests include computer vision and deep learning.

Xiu-Shen Wei (M'18) received his Ph.D. degree in computer science and technology from Nanjing University. He is a Professor at Nanjing University of Science and Technology (NJUST). Before joining NJUST, he served as the Founding Director of Megvii Research Nanjing, Megvii Technology. He has published more than thirty academic papers on the top-tier international journals and conferences, such as IEEE TPAMI, IEEE TIP, IEEE TNNLS, IEEE TKDE, Machine Learning, CVPR, ICCV, ECCV, IJCAI, ICDM, ACCV, etc. He won four world championships in international authoritative computer vision competitions, including iWildCam (in association with CVPR 2020), iNaturalist (in association with CVPR 2019), Apparent Personality Analysis (in association with ECCV 2016), etc. He also received the Presidential Special Scholarship (the highest honor for Ph.D. students) in Nanjing University, and received the Outstanding Reviewer Award in CVPR 2017. His research interests are computer vision and machine learning. He has served as a PC member of CVPR, ICCV, ECCV, NeurIPS, IJCAI, AAAI, etc. He is a member of the IEEE.

Bo-Rui Zhao received his BS and MS degrees in 2016 and 2019, at the Department of Electronic Science and Engineering of Nanjing University, China. He is currently a researcher with Megvii Research Nanjing. His research interests include computer vision, deep learning, and general object detection.



Yong-Shun Zhang received the BS degree from the Department of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, in 2019. He is currently a first year graduate student of School of Artificial Intelligence in Nanjing University. His research interests include computer vision and deep learning, especially focusing on long-tailed visual recognition and pruning of convolutional neural networks.

Xiaoyang Tan received the BS and MS degrees in computer applications from the Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, in 1993 and 1996, respectively, and the PhD degree from the Department of Computer Science and Technology, Nanjing University, Nanjing, in 2005. In 1996, he was an Assistant Lecturer with NUAA. From 2006 to 2007, he was a Post-Doctoral Researcher with the Learning and Recognition in Vision team, INRIA Rhone-Alpes, Grenoble, France. His current research interests include face recognition, machine learning, pattern recognition, and computer vision.

Yang Yu received the Ph.D. degree in Computer Science from Nanjing University in 2011, and then joined the LAMDA Group in the Department of Computer Science and Technology of Nanjing University as an Assistant Researcher from 2011, and as an Associate Professor from 2014. He joined the School of Artificial Intelligence of Nanjing University as a Professor from 2019. His research interest is in machine learning, and he founded Polixir Technologies Ltd. for landing reinforcement learning in real-world applications. He was a recipient of IEEE Intelligent Systems "AI's 10 to Watch" (2018), PAKDD Early Career Award (2018), the National Outstanding Doctoral Dissertation Award (2013), the China Computer Federation Outstanding Doctoral Dissertation Award (2011), etc.