Neurocomputing 430 (2021) 82–93

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Local-binarized very deep residual network for visual categorization

Xuejing Liu [a,b,1], Liang Li [a,*], Shuhui Wang [a], Zheng-Jun Zha [c], Qingming Huang [a,b]

[a] Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China
[b] School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100190, China
[c] School of Information Science and Technology, University of Science and Technology of China, Hefei 230027, China

ARTICLE INFO

Article history:
Received 23 December 2019
Revised 23 June 2020
Accepted 19 November 2020
Available online 2 December 2020
Communicated by Cheng Jun

Keywords:
Network compression and acceleration
Pose estimation
Object recognition
Saliency detection
Local binary residual block

ABSTRACT

Residual networks usually require more layers to achieve remarkable performance in complex visual categorization tasks, such as pose estimation. However, the increasing number of layers leads to a heavy burden on training and forward inference as well as over-fitting. This paper proposed local binary residual block (LBB) to promote the very deep residual networks on the trainable parameters, FLOPs and accuracy. In each LBB, the 3×3 filters are binarized based on Bernoulli distribution under a sparse constraint, an activation function is prepared to trigger the non-linear response, and the linear 1×1 filters are learned in a real-valued way. After stochastic binarized initialization, the 3×3 filters in LBB need not be updated during training. The above architecture reduces at least 69.2% trainable parameters and 70.5% FLOPs compared to the original model. The LBB is derived from three observations: 1) Activated responses of one standard $k \times k$ convolutional layer can be approximated by combining binarized $k \times k$ filters with 1×1 filters; 2) Most computation in the very deep residual networks is spent on the 3×3 convolutions; and 3) 1×1 filters play an important role in cross-channel information integration. In addition, the LBB module is suitable for the very deep network framework, including stacked hourglass network and pyramid residual modules. Experiments are conducted on MPII and LSP dataset for pose estimation task; CIFAR-10, CIFAR-100 and ImageNet datasets for object recognition; ECSSD, HKU-IS, PASCAL-S, DUT-OMRON, DUTS for saliency detection. The results show that our model can accelerate the training and inference of the network with only a slight performance degradation.

1. Introduction

Deep Residual Networks (ResNets) [1] have achieved state-of-the-art results in object detection [2] and localization [3]. In spite of the successes, ResNets require large amounts of memory and computational power for training due to the ever-increasing number of convolutional layers. In addition, ResNets show its impressive accuracy on many complex visual categorization tasks, such as human pose estimation [3,4].

To overcome the difficulties on pose estimation (occlusion, deformation, changes in clothing, lighting and camera views), the modified ResNets [3,5] can be up to 1000 layers, with a huge increase in the number of parameters and convolution operations compared to ordinary ResNets for classification. Such giant-scale network structures lead to a large number of training parameters, large time consumption and more data requirement for model training.

Along with the very deep networks, the need for inference speed-up is becoming urgent for real applications. For example, a lightweight and compact binary architecture [6] is designed by removing all the 1×1 convolutional filters for landmark localization. Experiments show that network parameter binarization greatly accelerates the inference time by replacing most arithmetic operations with bit-wise operations. However, compared with standard real-valued networks, the performance of such binarized network drops almost 12%, which is quite a large margin.

In deep networks, convolutional filters with different sizes capture different patterns in data. There are two main kinds of filters. First, **Spatial Convolution**, such as 3×3 or 5×5, has a receptive field capable of capturing locally spatial information. Its outputs indicate the response of the corresponding spatial patterns. It spends most computation sources on learning this kind of filter for generating more scalable spatial patterns. When the neural networks achieve very deep architecture with 1000 + layers, there are a large number of spatial filters. Even if our filters are randomly generated, the aggregation of responses from these filters can still

* Corresponding author.
E-mail address: liang.li@ict.ac.cn (L. Li).

[1] This work was supported in part by the National Key R&D Program of China under Grant 2018YFE0303104, in part by National Natural Science Foundation of China: 61732007, 61771457.

https://doi.org/10.1016/j.neucom.2020.11.041
0925-2312/

characterize rich and discriminant spatial patterns. Second, **1×1 Convolution** filters are created for three purposes, 1) cross-channel information integration of feature maps from a given layer; 2) dimension reduction in the filter space; 3) increasing non-linear activation layer like ReLU. A number of models accelerate the computation of the network by removing or binarizing the 1×1 filters. However, this has an adverse and irrecoverable influence on the whole network.

To reduce the deep model training complexity, we proposed *Local Binary Residual Block* (**LBB**), which is the basic module for very deep networks. In LBB, 3×3 spatial convolution filters are binarized, while 1×1 convolution filters are learned in real values. Different from existing work, the 3×3 filters in LBB are stochastically binarized under a sparse constraint of Bernoulli distribution, and such constraint brings rich spatial patterns to the filters. Furthermore, the LBB obtain an enormous amount of more discriminant spatial patterns by incrementally stacking the filters from different layers of very deep networks. In detail, weights of 3×3 filters in LBB only need to be initialized in the above way and do not need to be updated during training. Thus it reduced at least 69.2% trainable parameters for a 1000-layered ResNet due to the sparsity of binary filters, and it accelerated the whole training and inference stage because the binarized convolution operations are transformed into addition and subtraction operations. Besides, the experiments show the different initializations of 3×3 filters with Bernoulli distribution bring a slight fluctuation of performance.

The combination of 1×1 filters with binarized 3×3 filter responses aims to formulate the activated filter responses of one standard 3×3 convolutional layer. LBB can be seen as approximating the full-precision weight with the linear combination of binary weight bases. As Fig. 1b and Fig. 1c show, LBB is embedded into two different network blocks. One is the original block of stacked hourglass network (SHN), the other is the multi-branch pyramid residual module (PRM), and they are generally called *Local-Binarized Residual Networks* (LBRN). Due to its sparsity, LBRN can 1) save computations; 2) reduce the model size, both in terms of memory usage and disk space; 3) avoid overfitting. During inference, the convolution operation in 3×3 filters turns into judgment whether the information should be transferred or not to the next layer. It defines a new kind of convolution operation, which can significantly accelerate the forward propagation process.

The main contributions can be summarized as,

- We proposed local binary residual blocks (LBB) for accelerating the training and inference of very deep residual networks. LBB can be embedded in residual networks without bells and whistles. Promising results are obtained on the complex task, namely pose estimation, which indicates that it is not necessary to train all the parameters in the very deep networks.
- The spatial convolution filters in LBB are generated through stochastic binarization under a sparse constraint of Bernoulli distribution, whose weights do not need to be updated during training. This reduces at least 69.2% trainable parameters for a 1000-layered deep residual network. At inference time, the FLOPs of each block in LBRN is 0.5 GFLOPs while that of the full-precision network is 1.7 GFLOPs.
- Experiments show the LBB provides acceleration for the very deep network with only a small performance degradation. The performance is getting better as the number of layers and channels in 1×1 filters increases. The performance of the whole network is insensitive to the sparsity level setting of binary spatial filters.

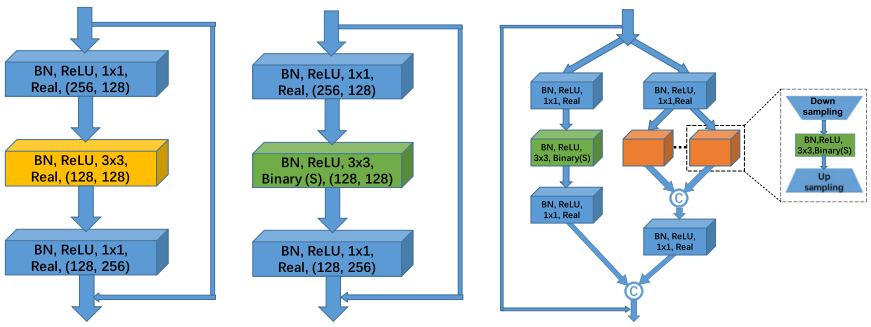
The rest of the paper is organized as follows. Section 2 presents related work on network compression and acceleration, especially binarization, and pose estimation. Section 3 introduces the methodology of our proposed LBB, and more technical details, such as the binarization method and network structure. Section 4 shows the quantitative experiments about our LBRN and their corresponding analysis. Section 5 draws the conclusions.

2. Related work

2.1. Binarization

A significant number of approaches [7–11] has been prompted to reduce the storage and computation costs of convolutional neural networks recently. These methods can be categorized into four directions: network pruning [12–14], network quantization [15–17], knowledge distillation [18–20] and compact network [21–23]. Particularly, binarization is a special case of quantization. Low precision binary network can save a lot of storage for DNNs. Besides, binary networks can only perform bit operations, which can greatly decrease the time consuming of DNNs. Thus approximating real-valued weights with binary representations has attracted much attention from researchers.

M. Courbariaux et al. introduce Binarized Neural Networks (BNNs), whose weights and activations is binarized at run-time. Thus the memory size and computation consuming are greatly decreased. Furthermore, it achieves almost the same performance of state-of-the-art results on several datasets [24]. M. Rastegari et al. propose two efficient networks: Binary-Weight-Networks



(a) (b) (c)

Fig. 1. The residual block structures of different ResNets. (a) The original bottleneck block of stacked hourglass networks. The parameters of the blocks are real-valued. (b) The structure of our LBB in stacked hourglass network. Only the 3×3 convolutions are binarized. (c) The structure of our LBB in pyramid residual module. Only the 3×3 convolutions are binarized. **Note:** a rectangular block containing: batch normalization, activation function, filter size, data type, the number of input and output channels.

and XNOR-Networks. Both the filters and input are binarize in XNOR, resulting in 58× faster convolutional operations and 32 × memory saving [25].

Based on XNOR, A. Bulat et al. design a lightweight, compact and suitable multi-scale residual network for landmark localization, which is the first work to apply binarization on landmark localization tasks [6]. F. Juefei-Xu et al. propose local binary convolutions, which couple pre-defined binary filters with learnable linear weights for classification [26]. Z. Li et al. introduce a high-order binarization scheme, which recursively performs residual quantization and got high-order binary filters and gradients. The recursive binarization operation can approximate the original network more accurately. [27]. X. Lin et al. propose ABC-Net, which is different from preceding networks in two aspects. Firstly, it approximates the full-precision weight with the linear combination of several binary weight bases. Secondly, it utilizes multiple binary activations to decrease information loss. ABC-Net achieves comparable results as full-precision networks [28]. A. Mishra and D. Marr improve the performance of low-precision networks through knowledge distillation. This approach is called Apprentice, which has three schemes and achieves state-of-the-art accuracy using ternary precision [29].

2.2. Pose estimation

A. Bulat and G. Tzimiropoulos design a detection-followed-by-regression CNN cascaded architecture to learn part relationship and spatial information, which is robust even for severe part occlusions [30]. SE. Wei et al. incorporate convolutional networks into pose machine framework to learn image features and image-dependent spatial models for pose estimation. They address the problem of vanishing gradients through intermediate supervision and get admirable results on several important benchmarks [31]. Z. Cao et al. propose a bottom-up parsing architecture to detect 2D pose of multiple people in an image, which achieves real-time high accuracy. This approach uses Part Affinity Fields, which is a non-parametric representation, to associate body parts in individuals. Their method exceeds previous results on MPII benchmark both in performance and efficiency [4]. Y. Chen et al. propose a structure-aware convolutional network, which incorporates prior knowledge of human bodies. They design discriminators to distinguish the real pose and the fake pose generated by pose generator to learn the priors successfully. Their approach outperforms the previous method and generates plausible human posed [32].

The very deep residual neural networks have achieved some remarkable results in many tasks, but they bring a huge burden on computation resources and have a huge number of parameters, which makes the networks difficult to train and slow to converge. Our model aims at optimizing the architecture of very deep neural networks for pose estimation with our local binary residual blocks (LBB), which can accelerate the procedure of network learning while keeping a slight performance degradation. We apply our LBB in two different networks for pose estimation: stacked hourglass network [3] and pyramid residual module [33]. Actually, our model can be easy to extend into any deep network structures. As each hourglass is a repeated bottom-up, top-down processing with intermediate supervision, it demands quite large computation resources. Our method is essentially a kind of trade-off between accuracy and time efficiency.

3. Methodology

In this section, we first introduce stacked hourglass network, pyramid residual modules, and local binary pattern. Stacked hourglass network is a widely used network for pose estimation with a repeated bottom-up, top-down structure. Pyramid residual modules aim to enhance the ability for multi-scale feature extraction of stacked hourglass network. Based on the two networks, we utilize local binary residual block (LBB) for pose estimation and get satisfying performance. Then we demonstrate how we apply LBB to the networks for pose estimation and the whole network structure. Next, we introduce two different binarization method and the reason why we use the stochastic binarization method under a sparse constraint in LBB. In the end, we formulate LBB and explain why it can decrease the inference and training time.

3.1. Stacked hourglass network

Stacked hourglass network (SHN) is proposed by Newell et al. [3], which is a widely used network for pose estimation. Due to the flexibility of poses and different points of view, the scale of different body parts in each image are changed tremendously. The motivation of SHN is to capture information of the image at every scale. The hourglass network fuses all the features together to output pixel-wise predictions.

SHN is a repeated bottom-up, top-down processing with an intermediate supervision structure, as shown in Fig. 2. The steps of pooling and subsequent is vital to combine features across all scales and capture multiple spatial relationships with the body, thus improving the performance of the network, and intermediate supervision is crucial for the network to get the considerable performance. Overall, their network shows robust results on MPII and FLIC benchmarks and outcompeting all the past methods.

3.2. Pyramid residual modules

The difficulty of pose estimation lies in scale variations of different parts due to camera view change and foreshortening. Although the structure of SHN can capture information of an image at every scale with its repeated bottom-up, top-down structure, in each block the SHN can only capture features at one scale. Thus Yang et al. propose pyramid residual modules (PRM) [33] to enhance the network's ability for scale variations. Each block in PRM is a multi-branch structure, as can be seen in Fig. 1c. The pyramid residual module is to build feature pyramids.

Yang et al. proposed four different kinds of pyramid residual modules to capture multi-scale features in an image. Our based model is shown in Fig. 1c. To capture multi-scale visual patterns or semantics, the convolutional network turns into multi-branch. The left branch is an identity mapping. The middle branch is the same structure as the block in SHN. The branch on the right is separated into multi-branches again. Each branch has filters with different subsampling ratios. To reduce the resolution slowly and subtly, fractional max-pooling [34] is adopted to resize the feature maps. The computation of fractional max-pooling is shown as follows:

$$s_c = 2^{-M\frac{c}{C}}, dffc = 0, \ldots, C, M \geqslant 1, \tag{1}$$

where s_c denotes the subsampling ratio between the output features and input features, which ranges from 2^{-M} to 1. c denotes the cth level. For example, when $c = 0$, the resolution does not change. When $M = 1, c = C$, the subsampling ration is $\frac{1}{2}$.

3.3. Network structure

Although SHN and PRM get remarkable results on pose estimation, the repeated residual blocks are quite a burden for training and testing. Taking SHN as an example, the number of its layers is up to 1464, which requires a large amount of computation and memory resources.

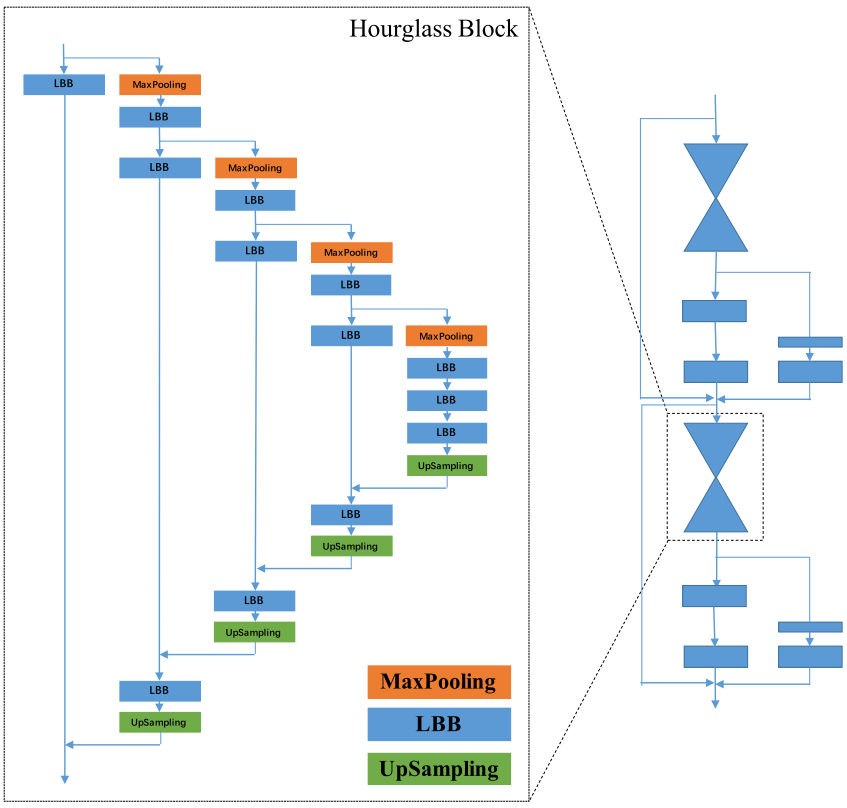


Fig. 2. The architecture of our LBRN under the structure of stacked hourglass networks, which stacks 8 hourglasses block. Each hourglass is composed of successive steps of pooling and upsampling. LBB is our designed local-binarized residual block, which is the core component block of LBRN.

To reduce time and memory consumption of residual networks for pose estimation, [6] proposed a full binarized residual network. The final block structure is a novel hierarchical, parallel, and multi-scale residual architecture, which is more suitable for binarization. Its binarization method is deterministic on the base of Binary-Weight-Network [25]. However, the performance of full binarized residual networks degrades significantly. In addition, a carefully designed residual block is vital to get better performance, which makes it less flexible to adapt to existing ResNets [1].

To alleviate the above problems, as illustrated in Fig. 2, we propose local binary residual block (LBB). In LBB, only the 3×3 filters are binarized under a sparse constraint while the 1×1 filters remain real-valued. The motivations of LBB stem from three aspects, 1) activated responses of one standard $k \times k$ convolutional layer can be approximated by 1×1 filters with binarized filter responses; 2) most computation in the very deep residual networks concentrates on the 3×3 convolutions; 3) 1×1 filters play an important part in cross-channel information integration.

LBB can be embedded in the structure of stacked hourglass network and pyramid residual module easily. As shown in Fig. 1a and b, LBB is able to keep the consistent structure as the original bottleneck of SHN. Fig. 1c shows the application of LBB to PRM, which doesn't change any of the original settings. LBB can be applied to residual networks without any bells and whistles. These residual networks with LBB for pose estimation are called LBRN for convenience in the subsequent.

As can be seen in Fig. 1b and c, the batch normalization and activation function precede a convolutional layer in LBB. All the 3×3 filters are binarized under a sparse constraint. Besides, in view of the effectiveness of the successive step of pooling and upsampling in SHN, the two LBRNs keep a similar design philosophy to it. At last, LBRNs produce a final set of heatmaps for prediction.

At the training stage, the weights of binarized filters do not need to be updated, so that there is no need to calculate the gradients in the 3×3 convolutional layers. For the original SHN, it can save at least 69.2% trainable parameters. For the task where filters with larger size are needed, LBRN help to reduce more trainable parameters. Taking 5×5 filters as an example, it can save 86.2% parameters during the training step. In testing, since the weights are binary, the convolution operations can be performed purely through additions and subtractions.

3.4. Binarization

In this subsection, we explore the binarization method in our local binary residual block (LBB) for better and faster processing.

Generally, two different binarization approaches are usually used in neural networks nowadays. One is **deterministic** methods as follows,

$$\text{sign}(x) = \begin{cases} +1 & x \geqslant 0, \\ -1 & x < 0, \end{cases} \tag{2}$$

The above sign function is a typical representation, where x denotes a real-valued variable. Based on this straightforward and intuitive function, several works have been proposed [35,25,28] to approximate the real-valued weights with binary weights.

However, shift-based batch normalization and specific updating rules of weights such as discretization of gradients and optimization methods are carefully designed in the deterministic binarization method. Furthermore, the weights are updated repeatedly during the whole training process, which needs a lot of training time.

The other one is the **stochastic** method, which requires the hardware to generate random bits to imitate the real-valued weights. It has two impressive advantages, 1) it is easy to implement as no specific updating rules are needed compared to deterministic ones 2) the binary weights need not be updated during the training stage once they are generated. These binarized layers are very similar to ReLU or Max Pooling layers, which do not have any learnable parameters. [26] applied randomly generated binary weights to standard convolutional neural networks (CNNs) and achieved performance comparable to regular CNNs on the task of classification. Considering the simplicity and efficiency, we design our LBB in the stochastic binarization method under a sparse constraint.

The procedure of stochastic binarization under a sparse constraint is as follows. First, a sparsity level is needed considering the tolerance of the weights to non-zero values. The sparsity level is set as 0.5 here, which means half of the parameters are zero. Second, the rest parameters are assigned to $+1$ or -1 randomly with equal probability, according to Bernoulli distribution. The Bernoulli random vectors are subgaussian and isotropic, which ensures that activated responses of one standard $k \times k$ convolutional layer can be approximated by 1×1 filters with binarized filter responses and more intermediate binary filters can result in a better approximation. Fig. 3 presents some examples of our binary convolutions.

3.5. Efficiency analysis

In this subsection, we take LBB based on stacked hourglass network as an example to explain why it can reduce trainable parameters and accelerate training and testing steps.

3.5.1. Residual Module

As Fig. 1b shows, LBB in an hourglass consists of three parts. For each residual module, let $X_l, X_{l+1} \in \mathbb{R}^{c\times w\times h}$ be the input and the output, where (c, w, h) represents channels, width, height respectively. let the intermediate results be denoted by $X_{m1}, X_{m2} \in \mathbb{R}^{c_m\times w\times h}$. $W_1 \in \mathbb{R}^{c\times 1\times 1\times c_m}, W_b \in \{0, \pm 1\}^{c_m\times 3\times 3\times c_m}, W_2 \in \mathbb{R}^{c_m\times 1\times 1\times c}$ represent the weights of these three different parts respectively, where the middle two dimensions are filter size and the other two are input and output channels. The forward procedure of LBB can be computed as follows:

$$\begin{cases} X_{m1} = \sigma_{\text{relu}}(X_l) \otimes W_1, \\ X_{m2} = \sigma_{\text{relu}}(X_{m1}) \oplus W_b, \\ X_{l+1} = \sigma_{\text{relu}}(X_{m2}) \otimes W_2 + X_l, \end{cases} \tag{3}$$

where $\otimes$ denotes convolution operation, $\oplus$ indicates a convolution without any multiplication, W_1, W_2 denotes real-valued weight, W_b indicates the binarized weights. The weights of 3×3 filters are binary thus we only need additions and subtractions.

3.5.2. Parameter Saving

The learnable parameters are greatly reduced as the 3×3 filters do not need to be updated during training. As we can see, the number of trainable parameters in original real-valued residual block can be computed as follows:

$$\mathrm{P}_r = c \times c_m + c_m \times 3 \times 3 \times c_m + c_m \times c. \tag{4}$$

The number of trainable parameters in LBB is as follows:

$$\mathrm{P}_b = c \times c_m + c_m \times c. \tag{5}$$

The ratio of the parameters of these two blocks is

$$\begin{aligned} \frac{\mathrm{P}_r}{\mathrm{P}_b} &= \frac{c\times c_m + c_m\times 3\times 3\times c_m + c_m\times c}{c\times c_m + c_m\times c} \\ &= 1 + \frac{9}{2} \times \frac{c_m}{c}. \end{aligned} \tag{6}$$

In our paper, c_m is usually half of c. So the ratio equals to 3.25. If we enlarge the filter size, for example, from 3×3 to 5×5, the ratio will be 7.25.

3.5.3. Computation efficiency

We compare the computation efficiency of LBB and real-valued residual block through floating point operations (FLOPs). The computation of FLOPs we used in our paper is the same as [36]. Convolution is regarded as a sliding window and the nonlinearity function is computed for free. Thus, the computation formulation is:

$$FLOPs = 2HW(C_{in}K^2 + 1)C_{out}, \tag{7}$$

where H and W are height and width, K is the kernal size and C_{in}, C_{out} denote the number of input and output channels.

Here, we omit the bias. Thus, for a real-valued residual block in stacked hourglass network the FLOPs is:

$$\begin{aligned} FLOPs_r &= 2 \times 2 \times 64 \times 64 \times 256 \times 1 \times 1 \times 128 + 2 \times 64 \times 64 \\ &\quad \times 128 \times 3 \times 3 \times 128 \\ &\approx 1.7 GFLOPs, \end{aligned} \tag{8}$$

For our LBB we compute FLOPs as:

$$\begin{aligned} FLOPs_b &= 2 \times 2 \times 64 \times 64 \times 256 \times 1 \times 1 \times 128 \\ &\approx 0.5 GFLOPs, \end{aligned} \tag{9}$$

so LBRN can be about $3\times$ faster than original real-valued networks.

3.6. Convergence analysis

Here is the proof why any 3×3 convolution can be approximated by 3×3 binary filter with 1×1 filters. Suppose X is the feature maps of input layer, B is the fixed binary $3 \times 3 \times M \times P$ filters, V is the $1 \times 1 \times P \times N$ filters, W is $3 \times 3 \times M \times N$ filters, M is the channel number of input layer, N is that of output layer, P is that of intermediate output). Our goal is to prove

$$C \approx B \tag{10}$$

when

$$\begin{aligned} &C = \sigma_{relu}(W^T X), \\ &B = D^T V, \\ &D\sigma_{relu}(BX). \end{aligned} \tag{11}$$

There are two cases. 1) $C = 0$: because $D >= 0$, there always exits a V such that $C = B$. 2) $C > 0$: If $D > 0$, there exits a V such that $C = B$. If $D = 0$, the approximation does not hold. According to Theorem. 5.39 in [37] and concentration inequalities, D vector are greater than zero with high probability. So any 3×3 convolution can be approximated by the cascading of 3×3 binary filter and 1×1 filters.

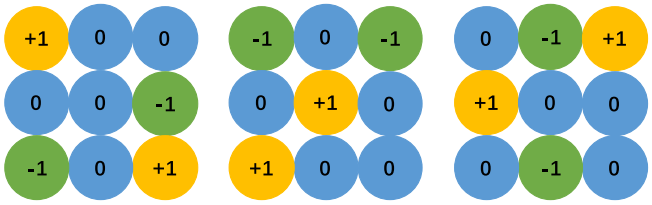


Fig. 3. Examples of our binary convolutions.

4. Experiments

We conduct experiments on two widely used datasets for human pose estimation: MPII, LSP and its extended dataset LSPET. We use the same PCKh evaluation method for both datasets. We test the two different LBRNs and compare the results with the original full-precision networks. Some experiments are conducted to verify the importance of 1×1 filters. Experiments on the sparsity level are presented. Ablation studies are provided to see the influence of different settings. We also conduct experiments on CIFAR-10, CIFAR-100 datasets and ImageNet for object recognition and ECSSD, HKU-IS, PASCAL-S, DUT-OMRON, DUTS for saliency detection. The results show that our method is suitable for object recognition and saliency detection as well.

4.1. Pose estimation on MPII dataset

The MPII human pose dataset, which is made for articulated human pose estimation [38]. It contains 25 K images over 40 K people with annotated body joints (28 k training and 11 k testing). Each image is extracted from YouTube video on every day human activities. Since testing annotations are not provided, all the results are generated on the standard training-validation partition of MPII. The validation contains about 3000 samples.

4.1.1. Training Settings

The LBRN based on stacked hourglass network (LBRN-S) is trained through RMSprob [39] algorithm with a learning rate of 2.5e-4, which can speed up first-order gradient descent methods. The training epoch is 100 with a mini-batch size of 4. Batch normalization operations are used before each convolutional layer, speeding up training and reducing the internal covariate shift of input data. The loss function is Mean Squared Error (MSE) criterion which minimizes the distance between prediction and target. Compared to the original stacked hourglass networks (SHN), the only extra hyper-parameter is sparsity level, the percentage of non-zero values in the local-binarized convolutional layers. It is notable that the weights of 3×3 convolutions are fixed so that there is no need to calculate the gradients or update the weights.

Data augmentation method such as scaling, rotation, flipping is used at training step for both networks. The LBRN based on PRM (LBRN-P) uses the same optimization method as stacked hourglass network. We train the PRM network with 2 stacked hourglasses. The training epoch is 250 with a mini-batch size of 6. The initialized learning rate is 2.5e-4 which is dropped by 10 at 150th, 170th and 200th epoch. The cardinality is set as 4, which means the feature pyramids will be divided into four scales.

4.1.2. Evaluation Measures

We take advantage of the Percentage Correct Keypoints (PCK) metric for evaluation. On MPII, PCKh metric, where the distance is normalized based on head size of the target [38], is used in our results. PCKh@0.5 denotes the distance less than a normalized distance of 0.5.

4.1.3. Results

We compare the LBRNs with real-valued ResNets, full-binarized ResNets [6] and full-binarized stacked hourglass network on the MPII dataset. Full-binarized stacked hourglass network has the same structure as our LBRN. Full-binarized ResNets [6] is a hierarchical, parallel and multi-scale structure designed for pose estimation. Fig. 1a, Fig. 1b and Fig. 1c denote the bottleneck block of stacked hourglass network, LBRN-S and LBRN-P respectively. We can see from the block structure that before each convolutional layer $(1 \times 1, 3 \times 3)$, batch normalization and ReLU layer are stacked. Both the LBRNs follow the structure of stacked hourglass as Fig. 2 shows. However, stacked hourglass network and its corresponding LBRN-S are built up with 8 hourglasses, while PRM and its corresponding LBRN-P have only 2 hourglasses. Table 1 presents the comparison results, and we can have the following findings.

First, comparing with full-binarized neural networks, LBRN-S and LBRN-P achieve a significant improvement of 9.3% and 9.5% separately while the size of parameters is very close. This means that our local binary residual module can boost the performance under a low parameter growing. It is not necessary to train all the parameters in a residual network for pose estimation.

Second, compared with SHN and PRM, LBRN-S and LBRN-P have a slight performance degradation of 1.4% and 0.8%, but the parameter number of LBRN-S and LBRN-P is about 30% of SHN and PRM, which means LBRN can save plenty of parameter space and training time. Furthermore, the 3×3 filters do not need to be updated during the training stage, and this can accelerate the training of the model. During inference time, the convolution operation in 3×3 filters turns into judgment whether the information should be transferred or not to the next layer. It defines a new kind of convolution operation, which can significantly accelerate the forward propagation process.

Third, due to the number of stacked hourglasses, the performance of PRM is less than SHN. However, the performance of LBRN-P is better than LBRN-S. Perhaps it is because of the multi-scale structure of PRM, which corresponds to different scales of local binary pattern.

Fourth, Fig. 5 shows some examples of pose estimation from these networks. The pose estimation of our LBRN is very similar to that of real-valued networks. On the contrary, many defects appear on the results of full-binarized residual networks. In short, our LBB can be regarded as one substitute for traditional very deep neural networks.

4.1.4. Ablation Study

We conducted experiments with different settings to find better network structure and hyper-parameters.

4.1.4.1. Sparsity Level. We first conducted experiments on different sparsity levels. Sparsity denotes the percentage of non-zero weights in a 3×3 filter. For example, the sparsity of 0.1 indicates that 90% of the weights in a 3×3 filter are zero. Table 2 present the results of experiments on different sparsity levels. It can be seen that the influence of different sparsity on performance is quite small, with a margin less than 0.5%. No matter how sparse the 3×3 is, the combination of 1×1 filters with binarized 3×3 filter responses can formulate the activated filter responses of one standard 3×3 convolutional layer well, which demonstrate the validity of this structure.

4.1.4.2. Importance of Convolutional Layers with 1×1 filter. In this subsection, we compare the performance of our LBRN with different numbers of 1×1 filters. We conduct several experiments to explore the impact of convolutional Layers with 1×1 filters. The only difference between the experiments is the number of 1×1 filters. Each LBB (Fig. 1b) of LBRN-S has two convolutional layers with 1×1 filters. We use 'LBB-1' to indicate the bottleneck block with only one 1×1 convolutional layer. Table 3 shows the detailed comparison, and we can find, the network with more 1×1 filters always get better performance. These three observations provide a strong support for the importance of 1×1 filters in residual networks. This is also one of the motivations of the local binarization in our LBRN. With these linear learnable parameters, binarized filter responses can be combined to approximate activated responses of one standard $k \times k$ convolutional layer easily.

Table 1
Results on MPII validation set (PCKh@0.5). HG-Binary: Full-binarized stacked hourglass network [6]. Binary: Full-binarized residual network [6]. SHN: Stacked hourglass network. LBRN-S: Local-binarized residual network based on stacked hourglass network. PRM: Pyramid residual module network. LBRN-P: Local-binarized residual network based on pyramid residual module. #Param: The amount of trainable parameters of each basic residual block.The Binary, SHN, LBRN-S are bult up with 8 hourglasses, while PRM, LBRN-P are built up with 2 hourglasses for efficiency. For LBRN-S on MPII dataset, the average PCKh over ten repetitions is 87.2+/-0.14.

	HG-Binary	Binary	SHN	LBRN-S	PRM	LBRN-P
Head	90.5	94.7	96.7	96.4	96.2	96.1
Shoulder	79.6	89.6	95.0	94.0	95.2	95.1
Elbow	63.0	78.8	88.2	87.5	89.2	88.2
Wrist	57.2	71.5	84.3	81.4	84.3	82.6
Hip	71.1	79.1	86.4	85.5	88.4	88.3
Knee	58.2	70.5	84.4	80.5	83.6	82.7
Ankle	53.4	64.0	80.7	77.0	80.6	79.2
Total	67.2	78.1	88.8	**87.4**	88.4	**87.6**
#Param	0.006	0.013 M	0.213 M	0.065 M	0.262 M	0.078 M
FLOPs	-	-	1.7G	0.5G	2.1G	0.6G

Table 2
Results on MPII validation set of different sparsity(PCKh@0.5).

Sparsity	Elbow	Wrist	Knee	Ankle	Total
0.1	86.6	81.6	81.8	78.2	87.2
0.2	86.4	81.9	80.8	78.1	87.0
0.3	87.1	82.2	81.3	77.8	87.4
0.4	87.0	82.2	81.3	77.8	87.4
0.5	86.7	81.1	81.3	78.4	87.1
0.6	87.2	81.4	81.0	78.2	87.2
0.7	87.5	81.9	81.2	77.6	87.4
0.8	86.8	81.5	81.4	78.6	87.1
0.9	87.2	81.6	81.7	77.8	87.2

Table 3
Results on MPII validation set (PCKh@0.5). The bold number denotes the number of 1×1 filters of each convolutional layer. 'LBRN' here denotes the networks based on stacked hourglass network (LBRN-S). There is only 1 layer of 1×1 filters in 'LBRN-1'. The sparsity here is 0.5.

	LBRN(**256**)	LBRN(**128**)	LBRN(**64**)	LBRN-1(**256**)
Head	96.4	96.0	95.3	95.8
Shoulder	94.0	93.7	92.5	92.9
Elbow	87.5	86.0	83.1	84.5
Wrist	81.4	80.1	76.6	78.2
Hip	85.5	85.4	83.1	82.4
Knee	80.5	80.5	76.6	77.8
Ankle	77.0	76.5	72.8	74.4
Total	87.1	86.5	84.1	84.8

4.1.4.3. Network Structure. We conduct experiments on three different network structure settings: 1) removing the 3×3 filters, 2) binarizing 1×1 convolutional layers while keeping 3×3 convolutional layers real-valued, 3) binarizing half the filters in 3×3 convolutional layers. Table 4 shows the results of the three different settings.

The total PCKh of LBRN-1 is 54.6%, which is far below the result (87.2%) of LBRN-S at sparsity level of 0.1. This is because 3×3 binary filters with a sparsity of 0.1 are different from 1×1 filters, even if there is only one activated element in a 3×3 filter. To generate one feature map, we need a $3 \times 3 \times N$ filter. The combination of N stacked 3×3 filters corresponds to a larger receptive field than 1×1 filters. It can bring more discriminative representation.

Most parameters and computation resources are spent on 3×3 convolutional layers. Binarizing 1×1 convolutional layers while keeping 3×3 convolutional layers real-valued is weak in network speedup and compression. The total PCKh of LBRN-2 is 86.7% on MPII dataset, which is 0.7% lower than LBRN-S. Besides, It spends $2.4\times$ longer time and $2.66\times$ memory than LBRN-S. The total PCKh of LBRN-3 is 88.4%, which is only 1% higher than LBRN-S. However, It consumes $2.2\times$ longer time and $2.1\times$ memory than LBRN-S.

Table 4
Results on MPII validation set (PCKh@0.5). LBRN-1 denotes the network removing all the 3×3 filters. LBRN-2 denotes the network binarizing 1×1 convolutional layers while keeping 3×3 convolutional layers real-valued. LBRN-3 denotes the network binarizing half the filters in 3×3 convolutional layers. The sparsity level here is 0.5.

	LBRN-1	LBRN-2	LBRN-3
Head	79.3	96.0	96.5
Shoulder	67.1	93.7	94.6
Wrist	48.5	80.4	83.8
Elbow	52.8	86.7	88.1
Hip	44.3	84.8	86.5
Knee	38.5	80.7	83.4
Ankle	40.6	77.7	79.8
Total	54.6	86.7	88.4

4.1.4.4. Network size. Two kinds of experiments are conducted to verify the impact of different model sizes. One is to alter the number of stacked hourglasses, the other one is to change the size of the hourglass module.

Table 5 shows the comparison results. We can observe, 1) networks with 8 stacked hourglasses outperform the ones with 4, which is consistent with the conclusion of real-valued ones [3]. 2) the performance descents if the hourglass is halved, which

Table 5
Results on MPII validation set (PCKh@0.5). The bold number denotes the number of stacked hourglasses. 'half' denotes the structure of hourglass is halved. The sparsity level here is 0.5.

	SHN(**8**)	SHN(**4**)	LBRN(**8**)	LBRN(**4**)	LBRN(**8**, half)
Head	96.6	96.4	96.2	95.7	96.4
Shoulder	95.2	94.2	94.2	93.4	93.4
Wrist	88.7	87.9	86.7	85.7	86.4
Elbow	84.1	82.9	81.1	80.2	80.9
Hip	87.0	86.1	85.5	85.3	85.0
Knee	83.6	82.6	81.3	80.8	80.0
Ankle	80.4	80.1	78.4	76.9	77.6
Total	88.8	88.0	87.1	86.5	86.6

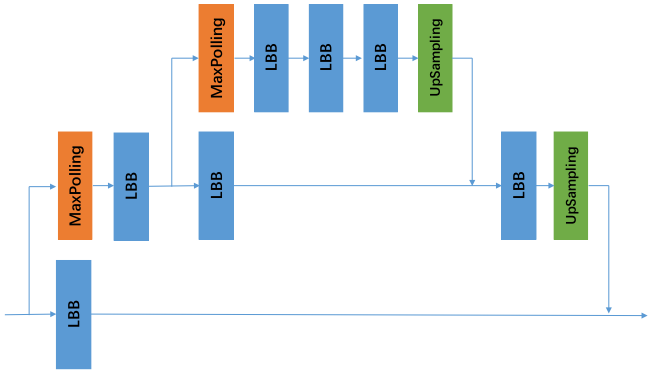


Fig. 4. Structure of halved hourglass (hg-half).

proves the effectiveness of the repeated pooling and upsampling hourglass structure. The structure of 'half' is shown in Fig. 4.

4.1.4.5. Activation function. Table 6 shows the result of different LBB with the sigmoid and ReLU activation functions Although the performance of LBRN-relu is better than that of LBRN-sigmoid, the difference is slight. This demonstrates that our LBB is universal to different activation functions.

4.2. Pose estimation on LSP and LSPET datasets

The Leeds Sports Pose (LSP) [40] and its extended training dataset [41], which contains 12000 images gathered from Flickr of mostly sportspeople. The images have been scaled such that the most prominent person is roughly 150 pixels in length. Each image has been annotated with up to 14 joint locations. The first 1000 images in LSP are used for training, and the last 1000 images are used for testing. The images in LSPET are all used for training. MPII training set is added to LSP and LSPET in the previous method. We follow the same setting to train our model.

4.2.1. Training settings

For the LSP test dataset, we use the image center as the body position and estimate the body scale by the image size. We augmented the training data through scaling, rotation, flipping, and adding color noise. The training settings are the same as the networks on MPII dataset.

4.2.2. Evaluation measures

We use PCKh metric to evaluate the results. Instead of PCKh@0.5, PCKh@0.2 is used for LSP dataset, which denotes the distance less than a normalized distance of 0.2.

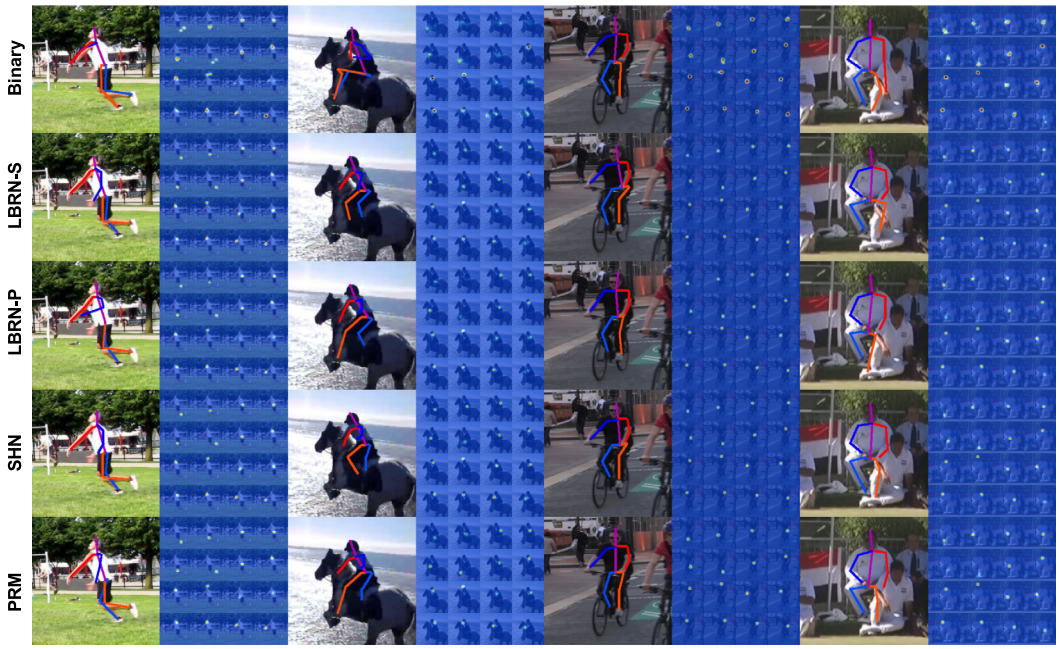


Fig. 5. The example output of pose estimation from different networks. The results of LBRN are similar to that of the full-precision network and outperform that of the full-binarized residual networks. On the left of each image, the final pose estimation is provided after the max activations across each heatmap. On the right we can see all the heatmaps of different joints.

Table 6
The performance comparison of the LBRN-S with different activation functions. The sparsity level here is 0.5.

	LBRN-relu	LBRN-sigmoid
Head	96.2	96.0
Shoulder	94.2	93.3
Elbow	86.7	86.1
Wrist	81.1	82.0
Hip	85.5	84.7
Knee	81.3	81.2
Ankle	78.4	77.4
Total	**87.1**	86.7

4.2.3. Results

The results of SHN, PRM and LBRN on LSP dataset are shown in Table 7. For SHN and LBRN-S, we stacked 8 hourglasses for training and testing, while PRM and LBRN-P are built up with 2 hourglasses for efficiency. That's the reason why SHN performs better than PRM. The results of SHN is 1.5% higher than LBRN-S and that of PRM is only 0.6% higher than LBRN-P. Similar to the results on MPII dataset, the margin between SHN and LBRN-S is larger than that of PRM and LBRN-P. This is corresponding to the characteristic of local binary pattern (LBP). It can meet the needs of capturing features of large scale structures when the LBP is multi-scale. Fig. 6 shows some examples and failure cases of the network output on LSP dataset.

4.2.4. Sparsity level

Table 2 shows different sparsity levels of LBRN-S have little impact on the performance for MPII dataset. We also conduct experiments of LBRN-P with different sparsity on LSP dataset. Table 8 shows the results. The same observation is obtained that the sparsity level has little impact on the performance, so we can choose the sparsity level according to the computing power of the hardware device. The representation ability of LBB depends on cascading binary 3×3 filters and 1×1 filters. It achieves the optima by learning the 1×1 filters in the training stage.

4.3. Object recognition on CIFAR datasets and ImageNet

The CIFAR Dataset [42], including CIFAR-10 and CIFAR-100, is a widely used dataset for object recognition. The CIFAR-10 dataset consists of 50000 training images and 10000 testing images, which are classified into 10 classes with 6000 images per class. Each image is a 32×32 color image so that it can be fast to train the dataset and test different algorithms. CIFAR-100 is kind of like CIFAR-10, containing 60000 images with a size of 32×32 each. The difference is that CIFAR-100 has 100 classes with 500 images for training and 100 for testing in each class. ImageNet (ILSVRC2012) has 1.2 M train images from 1 K categories and 50 K validation images. The images in this dataset are natural scenes with higher resolution. We evaluate our methods on a subset of it due to computational resource limitations.

4.3.1. Training settings

The ResNet we used for training contains 56 convolutional layers with three different bottlenecks. Each bottleneck has the structure of stacked $1 \times 1, 3 \times 3$ and 1×1 filters. The number of each bottleneck is 6. The network structure of ResNet-56 in our work is shown in Fig. 7.

Table 7
The performance comparison on LSP validation set (PCKh@0.2). SHN: Stacked hourglass network. LBRN-S: Local-binarized residual network based on stacked hourglass network. PRM: Pyramid residual module network. LBRN-P: Local-binarized residual network based on pyramid residual module. The SHN, LBRN-S are built up with 8 hourglasses, while PRM, LBRN-P are built up with 2 hourglasses for efficiency. The sparsity level os LBRN-S and LBRN-P is 0.5. For LBRN-P on LSP dataset, the average PCKh over ten repetitions is 91.9+/-0.41.

	SHN	LBRN-S	PRM	LBRN-P
Head	98.5	97.7	98.1	98.1
Shoulder	94.6	93.6	94.1	93.8
Elbow	90.2	88.7	89.9	88.6
Wrist	87.6	85.1	86.9	85.6
Hip	94.2	92.8	93.4	93.1
Knee	94.0	92.4	94.1	93.5
Ankle	93.5	91.5	92.8	92.2
Total	93.2	91.7	92.7	92.1

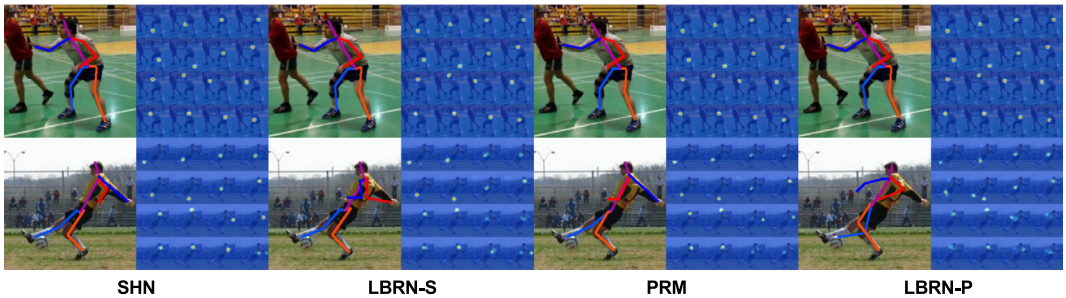


Fig. 6. Some example output of the different networks on LSP dataset. Some failure cases are shown in the second row. Our methods fail in distinguish the left and right joints.

Table 8
Results on LSP validation set of different sparsity (PCKh@0.5).

Sparsity	0.1	0.2	0.3	0.4	0.5
Total	91.1	91.9	91.8	91.6	92.1
Sparsity	0.6	0.7	0.8	0.9	1- 1
Total	92.0	91.9	91.9	92.3	1- 1

layer name	ResNet-56
conv1	3x3, 16
conv2_x	$6\times\begin{cases}1\times1, 16\\3\times3, 16\\1\times1, 64\end{cases}$
conv3_x	$6\times\begin{cases}1\times1, 32\\3\times3, 32\\1\times1, 128\end{cases}$
conv4_x	$6\times\begin{cases}1\times1, 64\\3\times3, 64\\1\times1, 256\end{cases}$
	average pool, fc, softmax

Fig. 7. The network structure of the ResNet-56 we used in our work.

4.3.2. Results

Our proposed LBB can be applied to ResNet with a little performance degradation, as Table 9 shows. ResNet-56 denotes the original full-precision ResNet, while the ResNet-LBB-56 denotes our local binary version. The results of ResNet-56 and ResNet-LBB-56 on CIFAR-10, CIFAR-100, and a subset of ImageNet are very close. Our method can save about 60% of the training parameters for the ResNet. During forward propagation, our method only costs half FLOPs of the original network.

4.4. Saliency detection on ECSSD, HKU-IS, PASCAL-S, DUT-OMRON and DUTS

Saliency detection [43–47] aims to detect the salient object in an image. With the development of residual networks, saliency detection has achieved surprising performance. To prove the effectiveness of our LBB on saliency detection, we use CPD [45] as our backbone and replace its original 3×3 filters with our local binary residual blocks. CPD is a cascaded partial decoder framework which discards low-level features and utilizes generated relatively precise attention map to refine high-level features to improve the performance. We evaluate our method on five benchmark datasets: ECSSD [48], HKU-IS [49], PASCAL-S [50], DUT-OMRON [51], DUTS [52]. We adopt mean absolute error (MAE) and F-measure (maxF) as our evaluation metrics.

4.4.1. Results

Table 10 shows the results of CPD and CPD-LB. CPD is the baseline method and CPD-LB denotes our local binarized network. The performance of CPD-LB is nearly the same as CPD. However, the flop of each basic convolution block is 9 times higher than the local binary residual block's. The results further validate the potential of our local binary residual block for visual categorization tasks.

5. Conclusion

Deep residual networks have achieved great success in many applications nowadays. With the increased layers, the computation resources become an impassable bottleneck. For complex tasks such as pose estimation, the depth of the networks can be up to 1000 layers. In this paper, we proposed a local binary residual block (LBB) to accelerate training and testing steps. LBB can be embedded into stacked hourglass network and pyramid residual module, which is called LBRN for simplicity. The LBB approximates the standard $k\times k$ convolutional layer by cascading binary $k\times k$ filters and 1×1 filters. The comparison experiments on MPII and LSP datasets show that LBB can reduce 69.2% parameters and 70.5% FLOPs with only a slight loss of accuracy. Compared to fully binarized residual networks, our approach can improve the perfor-

Table 9
The results of ResNet and ResNet-LBB on CIFAR-10, CIFAR-100 and a subset dataset of ImageNet. #Param: The number of trainable parameters. FLOPs: Floating point operations during inference time.

Dataset	Network	top-1	top-5	FLOPs	#Param
CIFAR-10	ResNet-56	7.09	0.26	0.069G	0.48 M
	ResNet-LBB-56	7.49	0.22	0.034G	0.19 M
CIFAR-100	ResNet-56	26.44	7.31	0.069G	0.48 M
	ResNet-LBB-56	27.21	7.69	0.034G	0.19 M
ImageNet-sub	ResNet-56	9.6	1.4	0.069G	0.48 M
	ResNet-LBB-56	9.8	1.6	0.034G	0.19 M

Table 10
Comparison of different methods on five benchmark datasets and two metrics including MAE (lower is better), max F-measure (higher is better). The comparison is under two settings (with VGG and ResNet50 backbone netowrk). CPD is the baseline method and CPD-LB represents CPD with local binary residual block.

Method	Backbone	ECSSD		HKU-IS		PASCAL-S		DUT-OMRON		DUTS-TEST	
		maxF	MAE	maxF	MAE	maxF	MAE	maxF	MAE	maxF	MAE
CPD	VGG16	0.936	0.040	0.924	0.033	0.866	0.074	0.794	0.057	0.864	0.043
CPD-LB	VGG16	0.933	0.041	0.924	0.033	0.851	0.075	0.779	0.058	0.858	0.045
CPD	ResNet50	0.939	0.037	0.925	0.034	0.864	0.072	0.797	0.056	0.865	0.043
CPD-LB	ResNet50	0.937	0.038	0.923	0.034	0.848	0.074	0.787	0.055	0.858	0.044

mance by 9.3%. Experiments also indicate the LBB is insensitive to the sparsity level. Besides, LBB can also be applied to object recognition and get good results. In the future, we will extend LBB to other deep architectures on more complex tasks.

CRediT authorship contribution statement

Xuejing Liu: Conceptualization, Methodology, Validation, Software. **Liang Li:** Conceptualization, Methodology. **Shuhui Wang:** Methodology. **Qingming Huang:** Resources, Funding acquisition.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: ict.ac.cn;ucas.ac.cn;ustc.edu.cn;vipl.ict.ac.cn.

References

[1] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27–30, 2016, 2016, pp. 770–778..

[2] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90.

[3] A. Newell, K. Yang, J. Deng, Stacked hourglass networks for human pose estimation, in: Computer Vision - ECCV 2016–14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VIII, 2016, pp. 483–499..

[4] Z. Cao, T. Simon, S. Wei, Y. Sheikh, Realtime multi-person 2d pose estimation using part affinity fields, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017, 2017, pp. 1302–1310..

[5] L. Ke, M. Chang, H. Qi, S. Lyu, Multi-scale structure-aware network for human pose estimation, CoRR abs/1803.09894..

[6] A. Bulat, G. Tzimiropoulos, Binarized convolutional landmark localizers for human pose estimation and face alignment with limited resources, CoRR abs/1703.00862..

[7] J.S. Smith, B. Wu, B.M. Wilamowski, Neural network training with levenberg-marquardt and adaptable weight compression, IEEE Trans. Neural Netw. Learning Syst. 30 (2) (2019) 580–587.

[8] J. Cheng, J. Wu, C. Leng, Y. Wang, Q. Hu, Quantized CNN: A unified approach to accelerate and compress convolutional networks, IEEE Trans. Neural Netw. Learning Syst. 29 (10) (2018) 4730–4743.

[9] J. Gui, Z. Sun, S. Ji, D. Tao, T. Tan, Feature selection based on structured sparsity: A comprehensive study, IEEE Trans. Neural Netw. Learning Syst. 28 (7) (2017) 1490–1507.

[10] N. Lee, T. Ajanthan, P.H.S. Torr, Snip: single-shot network pruning based on connection sensitivity, in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9, 2019, 2019..

[11] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K.T. Cheng, J. Sun, Metapruning: Meta learning for automatic neural network channel pruning, CoRR abs/1903.10258..

[12] S. Han, H. Mao, W.J. Dally, Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding, CoRR abs/1510.00149..

[13] Y. He, X. Zhang, J. Sun, Channel pruning for accelerating very deep neural networks, in: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22–29, 2017, 2017, pp. 1398–1406..

[14] Z. Zhuang, M. Tan, B. Zhuang, Y. JkkeyLiu, Q. Guo, J. Wu, J.Zh.u. Huang, Discrimination-aware channel pruning for deep neural networks, in: Advances in Neural Information Processing Systems 31 Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada, 2018, pp. 883–894.

[15] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, L.J.V. Gool, Soft-to-hard vector quantization for end-to-end learning compressible representations, in: , in: Advances in Neural Information Processing Systems 30 Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, 2017, pp. 1141–1151.

[16] S. Khoram, J. Li, Adaptive quantization of neural networks, in: International Conference on Learning Representations, 2018. https://openreview.net/forum?id=SyOK1Sg0W..

[17] R. Banner, I. Hubara, E. Hoffer, D. Soudry, Scalable methods for 8-bit training of neural networks, in: Advances in Neural Information Processing Systems 31 Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada, 2018, pp. 5151–5159.

[18] Q. Li, S. Jin, J. Yan, Mimicking very efficient network for object detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017, 2017, pp. 7341–7349..

[19] T. Chen, I.J. Goodfellow, J. Shlens, Net2net: Accelerating learning via knowledge transfer, CoRR abs/1511.05641..

[20] E.J. Crowley, G. Gray, A.J. Storkey, Moonshine: Distilling with cheap convolutions, in: Advances in Neural Information Processing Systems 31 Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada, 2018, pp. 2893–2903.

[21] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, CoRR abs/1704.04861..

[22] X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices, CoRR abs/1707.01083..

[23] H. Gao, Z. Wang, S. Ji, Channelnets: Compact and efficient convolutional neural networks via channel-wise convolutions, in: Advances in Neural Information Processing Systems 31 Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada, 2018, pp. 5203–5211.

[24] M. Courbariaux, Y. Bengio, Binarynet: Training deep neural networks with weights and activations constrained to +1 or -1, CoRR abs/1602.02830..

[25] M. Rastegari, V. Ordonez, J. Redmon, A. Farhadi, Xnor-net: Imagenet classification using binary convolutional neural networks, in: Computer Vision - ECCV 2016–14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV, 2016, pp. 525–542..

[26] F. Juefei-Xu, V.N. Boddeti, M. Savvides, Local binary convolutional neural networks, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017, 2017, pp. 4284–4293..

[27] Z. Li, B. Ni, W. Zhang, X. Yang, W. Gao, Performance guaranteed network acceleration via high-order residual quantization, in: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22–29, 2017, 2017, pp. 2603–2611..

[28] X. Lin, C. Zhao, W. Pan, Towards accurate binary convolutional neural network, in: Advances in Neural Information Processing Systems 30 Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, 2017, pp. 344–352.

[29] A.K. Mishra, D. Marr, Apprentice: Using knowledge distillation techniques to improve low-precision network accuracy, CoRR abs/1711.05852..

[30] A. Bulat, G. Tzimiropoulos, Human pose estimation via convolutional part heatmap regression, in: Computer Vision - ECCV 2016–14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII, 2016, pp. 717–732..

[31] S. Wei, V. Ramakrishna, T. Kanade, Y. Sheikh, Convolutional pose machines, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27–30, 2016, 2016, pp. 4724–4732..

[32] Y. Chen, C. Shen, X. Wei, L. Liu, J. Yang, Adversarial posenet: A structure-aware convolutional network for human pose estimation, in: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22–29, 2017, 2017, pp. 1221–1230..

[33] W. Yang, S. Li, W. Ouyang, H. Li, X. Wang, Learning feature pyramids for human pose estimation, in: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22–29, 2017, 2017, pp. 1290–1299..

[34] B. Graham, Fractional max-pooling, CoRR abs/1412.6071..

[35] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, Y. Bengio, Quantized neural networks: Training neural networks with low precision weights and activations, CoRR abs/1609.07061..

[36] P. Molchanov, S. Tyree, T. Karras, T. Aila, J. Kautz, Pruning convolutional neural networks for resource efficient transfer learning, CoRR abs/1611.06440..

[37] R. Vershynin, Introduction to the non-asymptotic analysis of random matrices, CoRR abs/1011.3027..

[38] M. Andriluka, L. Pishchulin, P.V. Gehler, B. Schiele, 2d human pose estimation: New benchmark and state of the art analysis, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23–28, 2014, 2014, pp. 3686–3693..

[39] T. Tieleman, G. Hinton, Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. (2012) 4(2)..

[40] S. Johnson, M. Everingham, Clustered pose and nonlinear appearance models for human pose estimation, in: Proceedings of the British Machine Vision Conference, 2010, doi:10.5244/C.24.12..

[41] S. Johnson, M. Everingham, Learning effective human pose estimation from inaccurate annotation, in: The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011, 2011, pp. 1465–1472..

[42] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images, Master's thesis, Department of Computer Science, University of Toronto..

[43] J. Han, D. Zhang, X. Hu, L. Guo, J. Ren, F. Wu, Background prior-based salient object detection via deep reconstruction residual, IEEE Trans. Circuits Syst. Video Techn. 25 (8) (2015) 1309–1321.

[44] Z. Wu, L. Su, Q. Huang, Stacked cross refinement network for edge-aware salient object detection, in, in: The IEEE International Conference on Computer Vision (ICCV), 2019.

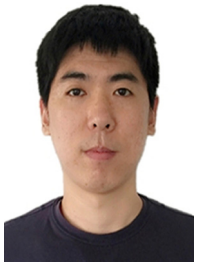
[45] Z. Wu, L. Su, Q. Huang, Cascaded partial decoder for fast and accurate salient object detection, in, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[46] D. Zhang, D. Meng, J. Han, Co-saliency detection via a self-paced multiple-instance learning framework, IEEE Trans. Pattern Anal. Mach. Intell. 39 (5) (2017) 865–878.

[47] D. Zhang, J. Han, Y. Zhang, D. Xu, Synthesizing supervision for learning deep saliency network without human annotation, IEEE Trans. Pattern Anal. Mach. Intell. 42 (7) (2020) 1755–1769.

[48] Q. Yan, L. Xu, J. Shi, J. Jia, Hierarchical saliency detection, 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013, 2013, pp. 1155–1162.
[49] G. Li, Y. Yu, Visual saliency based on multiscale deep features, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7–12, 2015, 2015, pp. 5455–5463..
[50] Y. Li, X. Hou, C. Koch, J.M. Rehg, A.L. Yuille, The secrets of salient object segmentation, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23–28, 2014, 2014, pp. 280–287..
[51] C. Yang, L. Zhang, H. Lu, X. Ruan, M. Yang, Saliency detection via graph-based manifold ranking, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013, 2013, pp. 3166–3173..
[52] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, X. Ruan, Learning to detect salient objects with image-level supervision, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017, 2017, pp. 3796–3805..

Xuejing Liu received the B.S. degree from Wuhan University in 2016. She is currently pursuing the Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences. She is also with the Key Laboratory of Intelligent Information Processing, Chinese Academy of Sciences. Her current research interests include machine learning, deep learning and computer vision.



Liang Li received his B.S. degree from Xi'an Jiaotong Univerisity in 2008, and Ph.D degree from Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China in 2013. From 2013 to 2015, he held a post-doc position with the Department of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing, China. Currently he is serving as the associate professor at Institute of Computing Technology, Chinese Academy of Sciences. He has also served on a number of committees of international journals and conferences, e.g. the GE of MTAP, the Organization Chair of ACM ICIMCS2015, the Special Session Chair of ACM ICIMCS2017 and IEEE GlobalSIP2017, and the Local Chair of PCM2017. Dr. Li has published over 30 refereed journal/conference papers, such as IEEE TNNLS, IEEE TMM, IJCAI, IEEE CVPR, DCC, etc. His research interests include image processing, large-scale image retrieval, image semantic understanding, multimedia content analysis, computer vision, and pattern recognition.

Shuhui Wang received the B.S. degree in electronics engineering from Tsinghua University, Beijing, China, in 2006, and the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2012. He is currently an Associate Professor with the Institute of Computing Technology, Chinese Academy of Sciences. He is also with the Key Laboratory of Intelligent Information Processing, Chinese Academy of Sciences. His current research interests include semantic image analysis, image and video retrieval and large-scale Web multimedia data mining.

Zhengjun Zha received the B.E. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2004 and 2009, respectively. He is currently a Full Professor with the School of Information Science and Technology, University of Science and Technology of China, the Vice Director of National Engineering Laboratory for Brain-Inspired Intelligence Technology and Application. He was a Researcher with the Hefei Institutes of Physical Science, Chinese Academy of Sciences, from 2013 to 2015, a Senior Research Fellow with the School of Computing, National University of Singapore (NUS), from 2011 to 2013, and a Research Fellow there from 2009 to 2010. He has authored or coauthired more than 100 papers in these areas with a series of publications on top journals and conferences. His research interests include multimedia analysis, retrieval and applications, as well as computer vision etc. Prof. Zha was the recipient of multiple paper awards from prestigious multimedia conferences, including the Best Paper Award and Best Student Paper Award in ACM Multimedia, etc.

Qingming Huang received the B.S. degree in computer science and the Ph.D. degree in computer engineering from the Harbin Institute of Technology, Harbin, China, in 1988 and 1994, respectively. He is currently a Professor and the Deputy Dean with the School of Computer and Control Engineering, University of Chinese Academy of Sciences. He has authored over 300 academic papers in international journals, such as the IEEE TPAMI, the IEEE TIP, the TKDE, the TMM and the TCSVT, and top level international conferences, including the ACM Multimedia, the ICCV, the CVPR, the ECCV, the VLDB, the AAAI, and the IJCAI. His current research interests include multimedia computing, image/video processing, pattern recognition, and computer vision.