A Lightweight Deep Network-Based Object Recognition Method Postprint

Authors: Li Yahui, Liu Jun

Date: 2019-01-03T00:00:00+00:00

Abstract

We propose a lightweight deep network-based approach for object recognition tasks under resource-constrained conditions. Through network architecture optimization techniques including convolution operation optimization, model parameter compression, and enhanced feature representation depth, we design and implement a lightweight network model structure tailored for embedded platform applications, which enables the deep network model to achieve substantial reductions in both model parameters and runtime resources while preserving accuracy. Experimental results demonstrate that the proposed lightweight deep model achieves 93.5% object recognition accuracy on the ImageNet-67 dataset when compressed to 10.2% of the baseline ResNet model proposed by the ILSVRC-15 champion.

Full Text

Preamble

Vol. 37 No. 3
Application Research of Computers –ChinaXiv Cooperative Journal
Accepted Paper

A Lightweight Deep Network-Based Object Recognition Method

Li Yahui, Liu Jun
(Fundamental Science on Communication Information Transmission & Fusion Technology Laboratory, Hangzhou Dianzi University, Hangzhou 310018, China)

Abstract: For object recognition tasks under resource-constrained conditions, this paper proposes a lightweight deep network-based object recognition method. By optimizing network structure through convolution operation refinement, model parameter compression, and enhanced feature expression depth, we design and implement a lightweight network architecture tailored for embedded platform applications. This enables deep network models to

achieve substantial reductions in parameters and resource requirements while maintaining accuracy. Experiments demonstrate that the proposed lightweight deep model, compared to the ResNet baseline model from the ILSVRC-15 champion, can compress the network model to 10.2% of the baseline size while retaining 93.5% object recognition accuracy on the ImageNet-67 dataset.

Keywords: deep learning; object recognition; lightweight; embedded application
Classification: TP183 **doi:** 10.19734/j.issn.1001-3695.2018.07.0663

0 Introduction

Automatic Target Recognition (ATR) in military applications serves as a critical component in battlefield environment perception. It utilizes various sensors to acquire target/background signals from the objective world and employs computer information processing methods to automatically analyze scene signals, detect and identify objects of interest, and obtain qualitative and quantitative properties of military targets. As a key technology in the spectral information domain, image target category detection technology extracts target features from raw images to complete recognition tasks, representing a fundamental problem in low-level computer vision processing. Also known as category-level object detection or simply object detection, this technology aims to detect target objects in images, determine their semantic categories, and localize them with bounding boxes using theories and methods from image processing and pattern recognition [1]. Image target category detection is a sub-problem of target classification, which can be divided into three levels: image-level (determining whether relevant targets exist in an image, such as image classification and annotation), region-level (determining whether a specific region contains a particular category—the focus of this paper), and pixel-level (determining which target class each pixel belongs to). Pixel-level can be further subdivided into category-level object segmentation and semantic segmentation. This paper focuses exclusively on the first level: image-level target recognition tasks.

With the rapid development of the Internet of Things (IoT) and big data, intelligent terminal devices play an increasingly important role in daily life, ushering in new opportunities for an era of intelligent perception and interconnected intelligence. This also provides foundational support for deep learning methods that heavily depend on hardware resources and data volume. Deep learning' s powerful feature extraction and representation capabilities have made it increasingly important in object detection, speech recognition, natural language processing, and other domains. Traditional computer vision tasks typically relied on features designed by domain experts for specific target categories—a process characterized by complexity and inefficiency that hindered cross-domain applicability. In contrast, deep learning automatically extracts target features through iterative weight updates during training, eliminating the cumbersome

manual feature design required by conventional algorithms. The emergence of transfer learning has further freed computer vision practitioners from intricate feature structure design, enabling rapid domain adaptation without network redesign.

However, deep learning algorithms impose higher demands on hardware computational capability and sample data quantity/quality. Deep network models, with their vast parameter counts and multi-level nonlinear mappings, exhibit excellent feature extraction and expression capabilities. Since feature extraction relies on iterative learning from sample sets, the process requires both sufficient hardware computational power and adequate dataset volume—two critical factors that enabled deep learning's resurgence in the big data era. Classic network models contain numerous learnable parameters, necessitating large datasets to ensure good generalization. Yet their intensive computational and storage requirements severely constrain application scope, limiting deployment primarily to high-performance server clusters.

Fortunately, recent research worldwide has revealed that while deep networks require numerous neurons to approximate complex nonlinear mappings, structural design factors introduce substantial parameter redundancy. Removing these redundant parameters does not significantly impact model performance. Therefore, this paper investigates lightweight network design for resource-constrained military applications such as UAVs. Based on the classical ResNet architecture, we optimize convolution operations, compress parameters, and enhance feature expression, achieving 93.5% recognition accuracy on ImageNet-67 while compressing the model to 10.2% of the original size.

[Figure 1: see original paper]

1 Related Work

Traditional computer vision tasks employed classical digital image processing methods, designing specialized feature extraction operators to capture specific target features, followed by classifiers like Support Vector Machines (SVM) for recognition. However, operator design requires expert domain knowledge and cannot be easily transferred across domains, slowing traditional computer vision development.

Since AlexNet' s five-layer deep network in 2012 opened new avenues for computer vision [2], subsequent models including ZF, VGG, GoogleNet, ResNet, DenseNet, and various optimized architectures have progressively improved performance. Model performance comparisons are shown in Figure 2 [Figure 2: see original paper] [3–16]. However, as performance improved, hardware requirements escalated from ordinary computers to large-scale high-performance server clusters, while datasets expanded from CIFAR-10/100 to ImageNet-1k and Open Images, growing from 60,000 to 1,900,000 images. Such large-scale intensive computation makes effective deployment on resource-constrained hardware challenging, prompting increasing research into deep network compression.

This field aims to substantially reduce model parameters and computations while preserving performance, thereby decreasing training and deployment requirements for sample data and hardware resources.

In 2016, MIT PhD student Song Han proposed a deep network compression method based on pruning, quantization, and Huffman coding, achieving AlexNet-level performance with models compressed to 1/510th the size. This sparked global research into network structure optimization for compression, yielding notable lightweight architectures for embedded applications such as MobileNet, ShuffleNet, and Google' s MobileNet v2 [8]. These models optimize neuron connectivity patterns from a structural design perspective, reducing parameters while maintaining performance.

In real industrial applications, especially IoT devices, hardware computational and storage capabilities differ dramatically from large-scale server clusters. Table 1 summarizes computational capabilities of common hardware devices.

Table 1. Common Graphics Card Specifications

Device	Processing Power (Tflops)
GeForce GTX 1080 GeForce GTX 1080 Ti Nvidia TITAN X Nvidia TITAN Xp GeForce GTX TITAN K80 GPU Accelerator Jetson Tegra K1	5.6~8.8

As shown, typical IoT devices offer only 3% of the computational power and 16.7% of the storage capacity of server-grade TITAN Xp hardware. Such severe resource constraints impose higher demands on complex deep network applications and make lightweight network research increasingly urgent.

2 Algorithm Framework

Deep network compression typically reduces model parameters and parameter storage space. However, simply removing structural parameters without optimization causes significant performance degradation, limiting practical utility. This paper addresses image-level target recognition. Drawing from champion ILSVRC network architectures and incorporating embedded application requirements, we designed a modular Se-DResNet network structure, shown in Figure 1.

Our deep network employs modular design, connecting modules according to topological structure to form the complete architecture. The Se-DResNet module structure is illustrated in Figure 3 [Figure 3: see original paper], with the

left diagram showing modules where input and output feature dimensions are identical, and the right showing dimension reduction cases.

The module processes input from the previous layer or data input layer by first splitting feature channels into two identical branches. One branch employs shortcut connections, directly passing to the module output precursor, while the other uses bottleneck principles to compress feature channels first (compression ratio configurable based on requirements). After bottleneck output, total feature channels are split according to a proportional coefficient and processed through different-scale convolution kernels to enrich receptive fields and enhance small network feature expression. The outputs are concatenated into a complete feature channel, followed by a channel-wise convolution operation for feature fusion across different receptive fields.

Given deep networks' substantial redundant features and varying feature quality across channels, we adopt a channel weighting approach that transforms each convolution channel into a scalar value measuring feature quality. This scalar serves as a channel weight for weighted operations across channels. The weighted features are then added element-wise through residual connections, outputting processed feature information to the next extraction module or classifier for target category identification.

Based on ResNet, our improvements target three aspects: convolution optimization, parameter compression, and enhanced feature expression, detailed below.

2.1 Optimizing Convolution Operations

Standard convolution computes the sum of input feature channels convolved with kernels as the next layer' s input, expressed in Equation (1):

$$f_s(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) \cdot g(x-s, y-t)$$

where w is the convolution kernel and g is the feature map.

While standard convolution extracts image features effectively, its computation requires kernel count determined by both input and output channels, creating strong coupling. Optimization from single direction proves difficult due to this coupling. We therefore employ depthwise separable convolution, dividing standard convolution into Depthwise and Pointwise components, as illustrated in Figure 4 [Figure 4: see original paper].

For standard convolution with input feature map dimensions $M \times D_k \times D_k$ (M input channels, D_k input size) and output dimensions $N \times D_f \times D_f$ (N output channels, D_f output size), computational cost is:

$$C_{std} = D_k \times D_k \times M \times N \times D_f \times D_f$$

Depthwise separable convolution maintains identical input/output dimensions while splitting computation into depthwise and pointwise components:

$$C_{dw} = D_k \times D_k \times M \times D_f \times D_f + M \times N \times D_f \times D_f$$

The reduction ratio compared to standard convolution is:

$$\gamma = \frac{C_{dw}}{C_{std}} = \frac{1}{N} + \frac{1}{D_k^2}$$

While 1×1 convolutions enable multi-channel information fusion, their pointwise operations remain computationally intensive despite fewer parameters. We adopt grouped convolution for all 1×1 operations, but omit channel shuffling for two reasons: (a) the first three grouped convolutions use different channel counts, inherently providing channel reordering and avoiding edge effects; (b) channel shuffling increases computation and memory usage, contradicting resource-constrained design requirements and degrading real-time performance.

2.2 Model Parameter Compression

To enrich receptive field diversity while reducing parameters, we employ bottleneck-based compression. Group convolutions first compress input 1×1 pointwise and 3×3 depthwise separable convolution channels using compression ratio w, then set input channel counts for 1×1 and 3×3 operations using ratio r. Both parameters can be selected based on accuracy and hardware requirements, as shown in Figure 5 [Figure 5: see original paper].

2.3 Enhanced Feature Expression

Lightweight networks for embedded applications suffer from insufficient feature expression capability, limiting performance compared to server-deployed architectures. To address this, we adopt SE-Net' s channel weighting mechanism to enhance feature expression, illustrated in Figure 6 [Figure 6: see original paper]. Unlike spatial dimension optimization (e.g., Inception), this approach optimizes channel relationships by explicitly modeling interdependencies and adaptively recalibrating channel-wise feature responses.

Specifically, original feature channels undergo global average pooling, followed by nonlinear activation to derive per-channel weight proportions. These weights are mapped back to each feature value in the original channels, producing weighted output feature maps. This coefficient-based weighting automatically differentiates features, emphasizing key target characteristics.

2.4 Reshaping the Loss Function

Dataset creation and preprocessing can introduce class imbalance and varying sample difficulty. Traditional cross-entropy loss fails to distinguish well-

recognized classes, causing the classifier to repeatedly learn already-mastered samples rather than focusing on difficult cases, increasing training time. We therefore adopt focal loss from RetinaNet, which uses predicted probabilities as loss weights. The focal loss expression is given in Equation (5):

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t)$$

where γ is the modulation coefficient controlling contribution rates of different samples (standard cross-entropy when $\gamma = 0$), and p_t is the predicted probability for the sample' s class during training.

3 Experiments

3.1 Experimental Environment

Our experimental environment comprises training and testing phases. Training uses a TITAN X platform with Xeon E5-2450@2.00 GHz CPU and 16 GB RAM, running Ubuntu 14.04 with the Caffe framework. Testing uses an NVIDIA Jetson TK1 platform with 2 GB onboard memory, running a customized Ubuntu 14.04 with Caffe.

3.2 Network Model Parameter Settings

Based on our lightweight deep network modules, we adjust model parameters for different hardware environments using $w = 1.0, r = 0.5$. Connecting modules forms the complete network structure, with detailed parameters in Table 2 (MB: Match_Block, DB: DRes_Block).

Table 2. Lightweight Model Structure Parameter Settings (ImageNet)

Layer	Output Size	Kernel Size	Stride	Repeat
Image	224×224	-	-	-
Conv1	112×112	-	-	-
Pool1	57×57	-	global	-
Pool2	57×57	-	-	-
···	29×29	-	-	-
···	15×15	-	-	-

The network begins with a standard convolution layer and pooling for image blurring to highlight target contours, followed by four "DB+MB"groups for high-level semantic feature extraction, and finally a softmax classifier for category identification.

Model training parameters are listed in Table 3 , using SGD optimization.

Table 3. Model Training Parameter Settings

Parameter	Value
Base Learning Rate	
Optimization Type	SGD
Learning Rate Policy	multistep
Maximum Iterations	
Learning Rate Gamma	

3.3 Experimental Results and Analysis

To rapidly validate feasibility and address military target recognition needs, we pruned the ImageNet-1k dataset by removing non-military categories, retaining 67 target classes including Person, Boat, etc. All experimental results below use this dataset for fair comparison.

We conducted two main comparisons: (a) original ResNet vs. our Se-DResNet, and (b) Se-DResNet vs. existing lightweight networks (MobileNet v1, MobileNet v2, ShuffleNet, ResNet-18).

Table 4 compares Se-DResNet with ResNet. "@.5" indicates channel pruning to 0.5× original width, etc.

Table 4. Model Size vs. Accuracy on ImageNet-67 Dataset

Model	Size (MB)	Compression Ratio (%)	Accuracy (%)	Error (%)
ResNet-50				
Se-DResNet	10.1	10.3%		
ResNet-50@.5		65.8%		
Se-DResNet@.5		16.7%		
ResNet-50@.25				
Se-DResNet@.25				

Without any pruning, Se-DResNet is only 10.1 MB—approximately 1/10th the size of ResNet-50—while performance differs by merely 0.9%. Under further compression, the model achieves higher accuracy than Se-DResNet@.5 at only 7.4% of the original structure. Memory usage during inference is also dramatically reduced: Se-DResNet requires only 533 MB, an 85% reduction from ResNet.

For embedded platform deployment, we performed channel-depth pruning on Se-DResNet to verify applicability across varying resource constraints. Table 5 shows that even at 1.7% of the original model size, performance remains effectively preserved, achieving accuracy comparable to ShuffleNet while occupying only 1.7 MB.

Table 5. Performance Comparison of Various Lightweight Network Models

Model	Size (MB)	Top-1 Accuracy (%)	Top-5 Accuracy (%)
ResNet-18	44.6		
ShuffleNet@1g3	7.3		
MobileNet	16.2		
MobileNet v2	14.2		
Se-DResNet	10.1		

Figure 7 [Figure 7: see original paper] compares convergence with ResNet-18, MobileNet v1/v2, and ShuffleNet. Se-DResNet demonstrates faster initial convergence and superior performance. In final training stages, all four lightweight models converge similarly, though ShuffleNet exhibits greater fluctuation and slightly lower performance than MobileNet variants. Se-DResNet achieves marginally higher accuracy than MobileNet v1 and matches MobileNet v2 (within 0.1%), while offering superior model size and convergence speed.

We conducted ablation studies to validate each improvement' s impact, shown in Table 6 .

Table 6. Influence of Ablation Studies on Classification Network Performance

Configuration	Depthwise Conv	Squeeze-Expand	Squeeze-Excitation	Focal Loss	Accuracy
Se-DResNet(1.0×)	√	√	√	√	

Using only depthwise separable convolution reduces performance significantly due to decreased cross-channel information interaction. Adding squeeze-expand structures with different kernel sizes enriches receptive field diversity, improving performance by 0.8%. Incorporating Squeeze-Excitation from SE-Net substantially boosts small model performance through enhanced feature expression. Focal loss improves performance by 1.7% by focusing training on hard samples.

4 Conclusion

This paper investigates lightweight network design for resource-constrained conditions, proposing the Se-DResNet architecture. Experiments demonstrate that Se-DResNet dramatically reduces parameters and computational intensity while preserving accuracy, enabling effective deployment on embedded platforms with limited hardware and energy resources. Currently, our approach focuses on image classification tasks and structural optimization only, without exploring

weight quantization, sharing, or encoding. Future work will integrate this lightweight structure with object detection frameworks and incorporate additional compression methods for embedded platform-based target detection tasks, providing effective reconnaissance capabilities and expanded battlefield perception for military applications.

References

[7] Huang Gao, Liu Zhuang, Weinberger K Q, et al. Densely connected convolutional networks [C]// Proc of IEEE conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE Press, 2017: 3.
[8] Iandola F N, Han Song, Moskewicz M W, et al. Squeezenet: alexnet-level accuracy with 50x fewer parameters and< 0.5 MB model size [EB/OL]. (2016) [2018-07-20]. https://arxiv.org/abs/1602.07360.
[9] Howard A G, Zhu Menglong, Chen Bo, et al. Mobilenets: efficient convolutional neural networks for mobile vision applications [EB/OL]. (2017) [2018-07-20]. https://arxiv.org/abs/1704.04861.pdf.
[10] Sandler M, Howard A, Zhu Menglong, et al. Inverted residuals and linear bottlenecks: mobile networks for classification, detection and segmentation [EB/OL]. (2018) [2018-07-20] https://arxiv.org/abs/1801.04381v2.pdf.
[11] Hu Jie, Shen Li, Sun Gang. Squeeze-and-excitation networks [EB/OL]. (2017) [2018-07-20]. https://arxiv.org/abs/1709.01507.pdf.
[12] Zhang Xiangyu, Zhou Xinyu, Lin Mengxiao, et al. ShuffleNet: an extremely efficient convolutional neural network for mobile devices [EB/OL] (2017) [2018-07-20]. https://arxiv.org/abs/1707.01083.pdf.
[13] Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift [EB/OL]. (2015) [2018-07-20]. https://arxiv.org/abs/1502.03167.pdf.
[14] Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, inception-resnet and the impact of residual connections on learning [C]// Proc of AAAI. Palo Alto, CA: AAAI Press, 2017: 12.
[15] Szegedy C, Vanhoucke V, Ioffe S, et al. Rethinking the inception architecture for computer vision [C]// Proc of IEEE conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE Press, 2016: 2818-2826.
[16] Chollet F. Xception: deep learning with depthwise separable convolutions [EB/OL]. (2016) [2018-07-20]. https://arxiv.org/abs/1610.02357.pdf.
[17] Xie Saining, Girshick R, Dollár P, et al. Aggregated residual transformations for deep neural networks [C]// Proc of Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE Press, 2017: 5987-5995.

Note: Figure translations are in progress. See original paper for figures.