Localization-aware Channel Pruning for Object Detection

Zihao Xie, Wenbing Tao*, Zhu Li, Lin Zhao
National Key Laboratory of Science and Technology on Multi-spectral Information Processing
School of Artificial Intelligence and Automation, Huazhong University of Science and Technology
Wuhan 430074, China

Abstract

Channel pruning is one of the important methods for deep model compression. Most of existing pruning methods mainly focus on classification. Few of them conduct systematic research on object detection. However, object detection is different from classification, which requires not only semantic information but also localization information. In this paper, based on DCP (Zhuang et al. 2018) which is state-of-the-art pruning method for classification, we propose a localization-aware auxiliary network to find out the channels with key information for classification and regression so that we can conduct channel pruning directly for object detection, which saves lots of time and computing resources. In order to capture the localization information, we first design the auxiliary network with a contextual ROIAlign layer which can obtain precise localization information of the default boxes by pixel alignment and enlarges the receptive fields of the default boxes when pruning shallow layers. Then, we construct a loss function for object detection task which tends to keep the channels that contain the key information for classification and regression. Extensive experiments demonstrate the effectiveness of our method. On MS COCO, we prune 70% parameters of the SSD based on ResNet-50 with modest accuracy drop, which outperforms the-state-of-art method.

Introduction

Since AlexNet (Krizhevsky, Sutskever, and Hinton 2012) won the ImageNet Challenge: ILSVRC 2012 (Russakovsky et al. 2015), deep convolutional neural network (CNNs) have been widely applied to various computer vision tasks, from basic image classification tasks (He et al. 2016) to some more advanced applications, e.g., object detection (Liu et al. 2016; Ren et al. 2015), semantic segmentation (Noh, Hong, and Han 2015), video analysis (Wang et al. 2016) and many others. In these fields, CNNs have achieved state-of-the-art performance compared with traditional methods based on manually designed visual features.

However, deep models often have a huge number of parameters and its size is very large, which incurs not only huge memory requirement but also unbearable computation burden. As a result, a typical deep model is hard to be deployed on resource constrained devices, e.g., mobile phones or embedded gadgets. To make CNNs available on resource-constrained devices, there are lots of studies on model compression, which aims to reduce the model redundancy without significant degeneration in performance. Channel pruning (He, Zhang, and Sun 2017; Luo, Wu, and Lin 2017; Jiang et al. 2018) is one of the important methods. Different from simply making sparse connections (Han, Mao, and Dally 2015; Han et al. 2015), channel pruning reduces the model size by directly removing redundant channels and can achieve fast inference without special software or hardware implementation.

In order to determine which channels to reserve, existing reconstruction-based methods (He, Zhang, and Sun 2017; Luo, Wu, and Lin 2017; Jiang et al. 2018) usually minimize the reconstruction error of feature maps between the original model and the pruned one. However, a well-reconstructed feature map may not be optimal for there is a gap between intermediate feature map and the performance of final output. Information redundancy channels could be mistakenly kept to minimize the reconstruction error of feature maps. To find the channels with true discriminative power for the network, DCP (Zhuang et al. 2018) attend to conduct channel selection by introducing additional discrimination-aware losses that are actually correlated with the final performance. It constructs the discrimination-aware losses by a fully connected layer which works on the entire feature map. However, the discrimination-aware loss of DCP is designed for classification task. Since object detection network uses the classification network as backbone, a simple method to conduct DCP for object detection is to fine-tune the pruned model, which was trained on classification dataset, for the object detection task. But the information that the two tasks need is not exactly the same. The classification task needs strong semantic information while what the object detection task needs is not only semantic information but also localization information. Hence, the existing training scheme may not be optimal due to the mismatched goals of feature learning for classification and object detection task.

In this paper, we propose a method called localization-

*Corresponding author

aware channel pruning (LCP), which conducts channel pruning directly for object detection. We propose a localization-aware auxiliary network for object detection task. First, we design the auxiliary network with a contextual roialign layer which can obtain precise localization information of the default boxes by pixel alignment and enlarges the receptive fields of the default boxes when pruning shallow layers. Then, we construct a loss function for object detection task which tends to keep the channels that contain the key information for classification and regression. Our main contributions are summarized as follows. (1) We propose a localization-aware auxiliary network which can find out the channels with key information so that we can conduct channel pruning directly on object detecion dataset, which saves lots of time and computing resources. (2) We propose a contextual ROIAlign layer which enlarges the receptive fields of the default boxes in shallow layers. (3) Extensive experiments on benchmark datasets show that the proposed method is theoretically reasonable and practically effective. For example, our method can prune 70% parameters of SSD (Liu et al. 2016) based on ResNet-50 (He et al. 2016) with modest accuracy drop on VOC2007, which outperforms the-state-of-art method.

Related Works

Network Quantization

Network quantization compresses the original network by reducing the number of bits required to represent each weight. Han et al. (Han, Mao, and Dally 2015) propose a complete deep network compression pipeline: First trim the unimportant connections and retrain the sparsely connected network. Weight sharing is then used to quantize the weight of the connection, and then the quantized weight and codebook are Huffman encoded to further reduce the compression ratio. Courbariaux et al. (Courbariaux et al. 2016) propose to accelerate the model by reducing the weight and accuracy of the output, because this will greatly reduce the memory size and access times of the network, and replace the arithmetic operator with a bit-wise operator. Li et al. (Li, Zhang, and Liu 2016) consider that multi-weights have better generalization capabilities than binarization and the distribution of weights is close to a combination of a normal distribution and a uniform distribution. Zhou et al. (Zhou et al. 2017) propose a method which can convert the full-precision CNN into a low-precision network, making the weights 0 or 2 without loss or even higher precision (shifting can be performed on embedded devices such as FPGAs).

Sparse or Low-rank Connections

Wen et al. (Wen et al. 2016) propose a learning method called Structured Sparsity Learning, which can learn a sparse structure to reduce computational cost, and the learned structural sparseness can be effectively accelerate for hardware. Guo et al. (Guo, Yao, and Chen 2016) propose a new network compression method, called dynamic network surgery, is to reduce network complexity through dynamic connection pruning. Unlike previous methods of greedy pruning, this approach integrates join stitching throughout the process to avoid incorrect trimming and maintenance of the network. Jin et al. (Jin et al. 2016) proposes to reduce the computational complexity of the model by training a sparsely high network. By adding a l_0 paradigm about weights to the loss function of the network, the sparsity of weights can be reduced.

Channel Pruning

Finding unimportant weights in the network has a long history. LeCun (LeCun, Denker, and Solla 1990) and Hassibi (Hassibi and Stork 1993) consider using the Hessian, which contains second order derivative, performs better than using the magnitude of the weights. Computing the Hessian is expensive and thus is not widely used. Han (Han, Mao, and Dally 2015) et al. proposed an iterative pruning method to remove the redundancy in deep models. Their main insight is that small-weight connectivity below a threshold should be discarded. In practice, this can be aided by applying l_1 or l_2 regularization to push connectivity values to become smaller. The major weakness of this strategy is the loss of universality and flexibility, thus seems to be less practical in real applications. Li et al. (Li et al. 2016) measure the importance of channels by calculating the sum of absolute values of weights. Hu et al. (Hu et al. 2016) define average percentage of zeros (APoZ) to measure the activation of neurons. Neurons with higher values of APoZ are considered more redundant in the network. With a sparsity regularizer in the objective function (Alvarez and Salzmann 2016; Liu et al. 2017), training based methods are proposed to learn the compact models in the training phase. With the consideration of efficiency, reconstruction-methods (He, Zhang, and Sun 2017; Luo, Wu, and Lin 2017) transform the channel selection problem into the optimization of reconstruction error and solve it by a greedy algorithm or LASSO regression. DCP (Zhuang et al. 2018) aimed at selecting the most discriminative channels for each layer by considering both the reconstruction error and the discrimination-aware loss.

Proposed Method

The auxiliary network we propose mainly consists of two parts. First, a contextual ROIAlign layer is designed to extract the features of the boxes. Then, a loss is designed for object detection task which can reserve the important channels. After the auxiliary network is constructed, we conduct channel pruning with the localization-aware losses of the auxiliary network. Fig. 1 is the overall frame diagram. The details of the proposed approach are elaborated below.

Contextual ROIAlign Layer

For object detection task, if we predict the bounding boxes directly on the entire feature maps, there will be a huge amount of parameters and unnecessary noises. So, it is important to extract the feature of region of interest (RoI) , which can be better used for classification and regression. To obtain precise localization information and find out the channels which are important for classification and regression, RoIAlign layer is a good choice which properly align the extracted features with the input. RoIAlign use bilinear

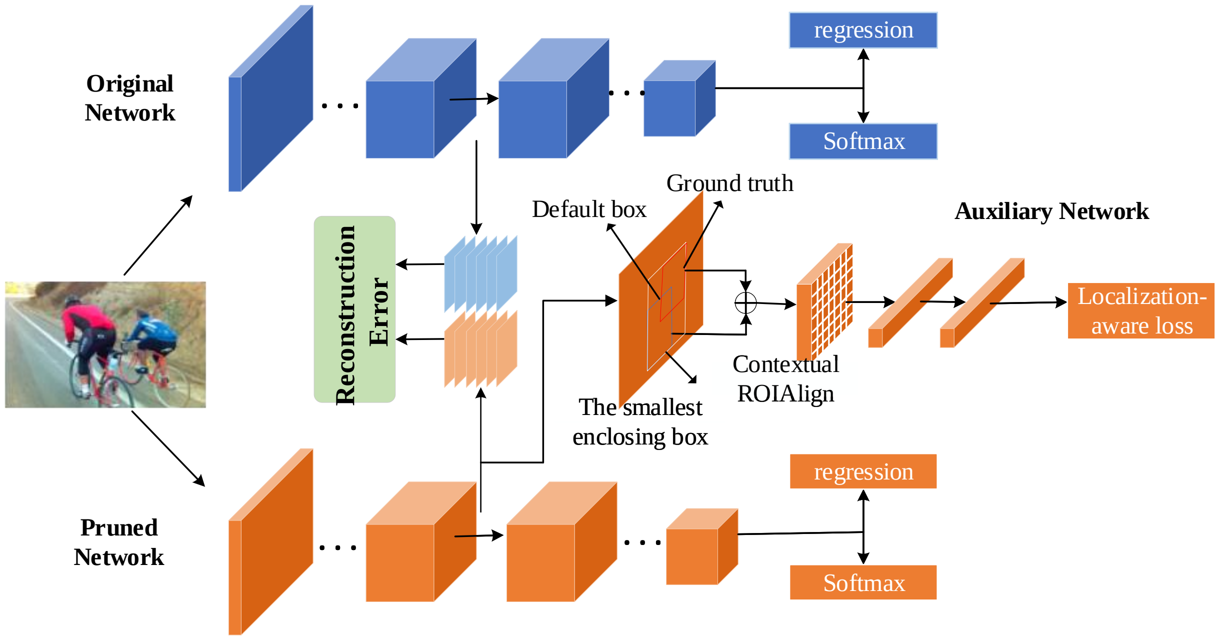

Figure 1: Illustration of localization-aware channel pruning. The auxiliary network is used to supervise layer-wise channel selection.

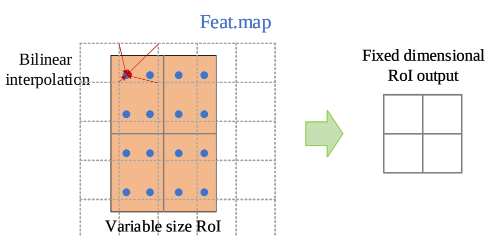

Figure 2: RoIAlign: The dashed grid represents a feature map, the solid lines an RoI (with $2\times$ bins in this example), and the dots the 4 sampling points in each bin. RoIAlign computes the value of each sampling point by bilinear interpolation from the nearby grid points on the feature map.

interpolation to compute the exact values of the input features at four regularly sampled locations in each RoI bin, and aggregate the result (using max or average), see Fig. 2 for details. However, the default boxes generated by the detector do not always completely cover the object area. From Fig. 3, we know that, the defalut box is sometimes bigger than the ground truth and sometimes smaller than it. So, the receptive fields may be insufficient if we only extract the features of the default box especially when we prune the shallow layers. To solve this problem, we propose a contextual ROIAlign layer, which introduces larger context information. The details are discussed below.

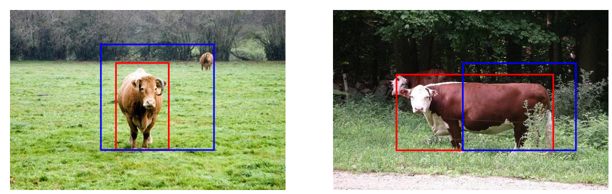

Figure 3: The features of default boxes do not always contain enough context information, especially when we prune shallow layers. The blue is default box, the red is ground truth.

For better description of the algorithm, some notations are given first. For a training sample, $(x_{a1}, y_{a1}, x_{a2}, y_{a2})$ represents the coordinates of ground truth box A, $(x_{b1}, y_{b1}, x_{b2}, y_{b2})$ denotes the coordinates of the matched default box B. We further use $\mathcal{F}$ to denote the feature map and $\mathcal{F}_S$ represents the features of area S, $ROIAlign$ represents the ROIAlign operation. First, we calculate the IoU of box A and B:

$$IoU_{AB} = \frac{A \cap B}{A \cup B} \quad (1)$$

B is a positive sample only if IoU_{AB} is larger than a preset threshold. We do not conduct contextual ROIAlign for B when B is negative sample. If B is a positive sample, then we calculate the smallest enclosing convex object C for A

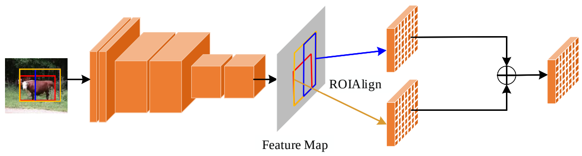

Figure 4: Contextual RoIAlign: The red is ground truth, the blue is default box, the yellow is the smallest enclosing box.

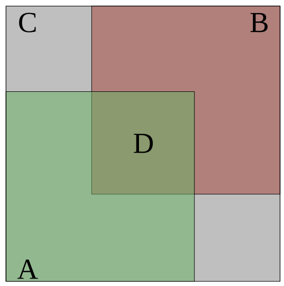

Figure 5: A, B are two arbitrary shapes, C is the smallest enclosing convex of A and B, D is the $[IoU]$ of A and B.

and B:

$$x_{c1} = min(x_{a1}, x_{b1}) \tag{2}$$

$$y_{c1} = min(y_{a1}, y_{b1}) \tag{3}$$

$$x_{c2} = max(x_{a2}, x_{b2}) \tag{4}$$

$$y_{c2} = max(y_{a2}, y_{b2}) \tag{5}$$

where $(x_{c1}, y_{c1}, x_{c2}, y_{c2})$ are the coordinates of C. Finally, the output of contextual ROIAlign layer is defined as:

$$\mathcal{F}_{\mathcal{O}} = ROIAlign(\mathcal{F}_{\mathcal{B}}) + ROIAlign(\mathcal{F}_{\mathcal{C}}) \tag{6}$$

Now we can get the precise features of default box B, the process can refer to Fig 4.

Construction of the Loss for Channel Pruning

After we construct the contextual ROIAlign layer, we need to consturct a loss for object detection task so that we can use the gradient of the auxiliary network to conduct model pruning. The details are discussed below.

In the stage of channel pruning, we use cross entropy and GIoU (Rezatofighi et al. 2019) to construct the loss of the auxiliary network. It is reasonable to use GIoU as loss function for boxes regression. It considers not only overlapping areas but also non-overlapping areas, which better reflects the overlap of the boxes. The GIoU of two arbitrary shapes A and B is defined as:

$$GIoU_{AB} = IoU_{AB} - \frac{C-U}{C} \tag{7}$$

where $U = A + B - IoU_{AB}$, IoU_{AB} and C are calculated by Eq. 1 - Eq. 5. Fig. 5 is a schematic diagram of GIoU. Then, we use G_i to denote the GIoU of the i-th predicted box and the ground truth, E_i to represent the cross entropy of the i-th predicted box. Then, in the pruning stage, $\mathcal{L}_{ac}$ represents the classification loss, $\mathcal{L}_{ar}$ represents the regression loss, $\mathcal{L}_{ap}$ represents the localization-aware loss of the auxiliary network. Finally, the loss of positive samples in pruning stage is defined as:

$$\mathcal{L}_{ac} = \sum_i E_i \tag{8}$$

$$\mathcal{L}_{ar} = \sum_i m(1 - G_i) \tag{9}$$

$$\mathcal{L}_a = \mathcal{L}_{ac} + \mathcal{L}_{ar} \tag{10}$$

where m is a constant coefficient.

Localization-aware Channel Pruning

After we construct the auxiliary network and the localization-aware loss, we can conduct channel pruning with them layer by layer. The pruning process of the whole model is described in Algorithm 1. For better description of the channel selection algorithm, some notations are given first. Considering a L layers of the CNN model and we are pruning the l-th layer, X represents the output feature map of the lth layer, W denotes the convolution filter of the $(l+1)$-th layer of the pruned model and $*$ represents the convolution operation. We further use $F \in R^{N \times HY}$ to denote output feature maps of the $(l+1)$-th layer of the original model. Here, N, H, Y represents the number of output channels, the height and the width of the feature maps respectively. Finally we use $\mathcal{L}_c$ and $\mathcal{L}_r$ to denote classification loss and regression loss of the pruned network.

To find out the channels which really contribute to the network, we should fine-tune the auxiliary network and pruned network first and the fine-tune loss is defined as the sum of the losses of them:

$$\mathcal{L}_f = \mathcal{L}_a + \mathcal{L}_c + \mathcal{L}_r \tag{11}$$

In order to minimizing the reconstruction error of a layer, we introduce a reconstruction loss as DCP does which can be defined as the Euclidean distance of feature maps between the original model and the pruned one:

$$\mathcal{L}_{re} = \frac{1}{2Q} \|F - X * W_{\mathcal{C}}\|_2^2 \tag{12}$$

where $Q = M \times H \times Y$, $\mathcal{C}$ represents the selected channels, $W_{\mathcal{C}}$ represents the submatrix indexed by $\mathcal{C}$.

Taking into account the reconstruction error, the localization-aware loss of the auxiliary network, the problem of channel pruning can be formulated to minimize the following joint loss function:

$$\begin{aligned} \min_{W_{\mathcal{C}}} \quad \mathcal{L}(W_{\mathcal{C}}) =& \mathcal{L}_{re}(W_{\mathcal{C}}) + \alpha \mathcal{L}_a(W_{\mathcal{C}}) \\ s.t. \quad & \|\mathcal{C}\|_0 \leq \mathcal{K} \end{aligned} \tag{13}$$

Algorithm 1 The proposed method

Input: number of layers L, weights of original model $\{W^l : 0 < l < L\}$, the training set $\{x_i, y_i\}$, the pruning rate η.

Output: $\{W_{\mathcal{C}}^l : 0 < l < L\}$: weights of the pruned model.

1: Initialize $W_{\mathcal{C}}^l$ with W^l for $\forall 1 \leq l \leq L$
2: **for** $l = 1, 2, \cdots, L$ **do**
3: Construct the fine-tune loss $\mathcal{L}_f$ shown as in Eq. 11
4: Fine-tune the auxiliary network and the pruned model by $\mathcal{L}_f$
5: Construct the joint loss $\mathcal{L}$ shown as in Eq. 13
6: Conduct channel selection for layer l by Eq. 14
7: Update $W_{\mathcal{C}}^l$ w.r.t. the selected channels by Eq. 15
8: **end for**
9: return the pruned model

where α is a constant, $\mathcal{K}$ is the number of channels to be selected. Directly optimizing Eq. 13 is NP-hard. Following general greedy methods in DCP, we conduct channel pruning by considering the gradient of Eq. 13. Specifically, the importance of the k-th channel is defined as:

$$\mathcal{S}_k = \sum_{i=1}^{H} \sum_{j=1}^{W} \| \frac{\partial \mathcal{L}}{\partial W_{k,i,j}} \|_2^2 \tag{14}$$

$\mathcal{S}_k$ is the square sum of gradient of the k-th channel. Then we reserve the channels with the i largest importance and remove others. After this, the selected channels is further optimized by stochastic gradient (SGD). $W_{\mathcal{C}}$ is updated by:

$$W_{\mathcal{C}} = W_{\mathcal{C}} - \gamma \frac{\partial \mathcal{L}}{\partial W_{\mathcal{C}}} \tag{15}$$

where γ represents the learning rate. After updating $W_{\mathcal{C}}$, the channel pruning of a single layer is finished.

Experiments

We evaluate LCP on the popular 2D object detector SSD (Liu et al. 2016). Several state-of-the-art methods are adopted as the baselines, including ThiNet and DCP. In order to verify the effectiveness of our method, we use VGG and ResNet to extract feature respectively.

Dataset and Evaluation

The results of all baselines are reported on standard object detection benchmarks, i.e. the PASCAL VOC (Everingham et al. 2010) . **PASCAL VOC2007 and 2012:** The Pascal Visual Object Classes (VOC) benchmark is one of the most widely used datasets for classification, object detection and semantic segmentation. We use the union of VOC2007 and VOC2012 trainval as training set, which contains 16551 images and objects from 20 pre-defined categories annotated with bounding boxes. And we use the VOC2007 test as test set which contains 4592 images. In order to verify the effectiveness of our method, on PASCAL VOC, we first compare our method only with ThiNet based on VGG-16 because the authors of DCP do not release the VGG model. To this end, we compare our method with DCP and ThiNet based on ResNet-50. Then we conduct the ablation experiment of our method on PASCAL VOC. In order to more fully verify the effectiveness of our method, we also perform experiments on the MS COCO2017 dataset.

In this paper, we use $07metric$ for all experiments on PASCAL VOC. For experiments on MS COCO, the main performance measure used in this benchmark is shown by AP, which is averaging mAP across different value of IoU thresholds, i.e. $IoU = \{.5, .55, \cdots, .95\}$.

Implementation details

Our experiments are based on SSD and the input size of the SSD is 300×300. We use VGGNet and ResNet as the feature extraction network for experiments. For ThiNet, we implement it for object detection. And the three methods prune the same number of channels for each layer. Other common parameters are described in detail below.

For VGGNet (Simonyan and Zisserman 2014), we use VGG-16 without Batch Normalization layer and prune the SSD from conv1-1 to conv5-3. The network is fine-tuned for 10 epochs every time a layer is pruned and the learning rate is started at 0.001 and divided by 10 at epoch 5. After the model is pruned, we fine-tune it for 60k iterations and the learning rate is started at 0.0005 and divided by 10 at iteration 30k and 45k, respectively.

For ResNet (He et al. 2016), we use the layers of ResNet-50 from conv1-x to conv4-x for feature extracting. The network is fine-tuned for 15 epochs every time a layer is pruned and the learning rate is started at 0.001 and divided by 10 at epoch 5 and 10, respectively. After the model is pruned, we fine-tune it for 120k iterations and the learning rate is started at 0.001 and divided by 10 at iteration 80k and 100k, respectively.

For the loss of auxiliary network, we set m to 50.

Experiments on PASCAL VOC

On PASCAL VOC, we prune the VGG-16 from conv1-1 to conv5-3 with compression ratio 0.75, which is 4x faster. We report the results in Tab. 1. From the results, we can see that our method achieves the best performance under the same acceleration rate. The accuracy of reconstruction based method like ThiNet drops a lot. But for our LCP, there is not much degradation in the performance of object detection. It is proved that our method retain the channels which really contribute to the final performance. Then we conduct the experiment based ResNet-50. We report the results in Tab. 2. From the results, LCP achieves the best performance regardless of pruning by 75% or pruning by 50%, which proves that our method can reserve the channels which contain key information for classification and regression. In addition, the ThiNet outperforms the DCP when pruning ratio is 0.7, which indicates that pruning the model on classification dataset for object detection is not optimal.

Experiments on MS COCO

In this section, we prune the ResNet-50 by 70% on COCO2017. We report the results in Tab. 3 and Tab. 4.

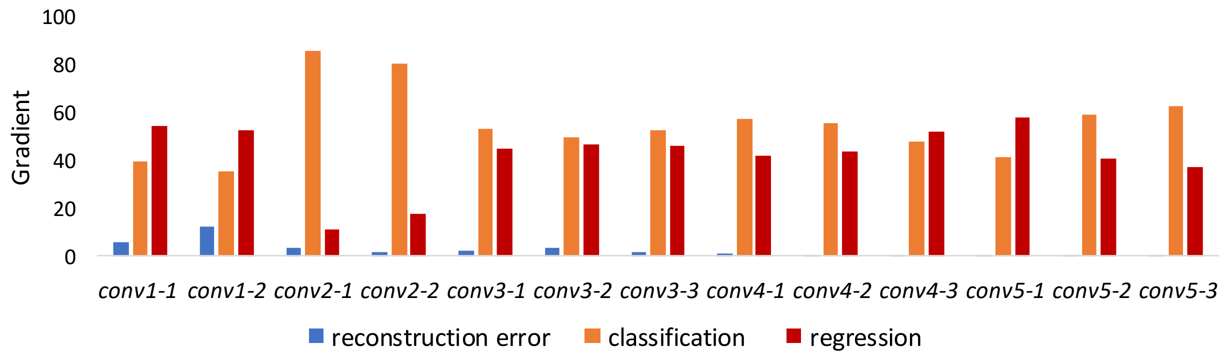

Figure 6: The percentage of the gradients generated by the three loss functions.

Table 1: The pruning results on PASCAL VOC2007. We conduct channel pruning from conv1-1 to conv5-3.

Method	backbone	η	flops↓	params↓	mAP
Original	VGG-16	0	0	0	77.4
ThiNet	VGG-16	0.5	50%	50%	74.6
LCP(our)	VGG-16	0.5	50%	50%	**77.2**
ThiNet	VGG-16	0.75	75%	75%	72.7
LCP(our)	VGG-16	0.75	75%	75%	**75.2**

Table 2: The pruning results on PASCAL VOC2007. We conduct channel pruning from conv2-x to conv4-x.

Method	backbone	η	flops↓	params↓	mAP
Original	ResNet-50	0	0	0	73.7
DCP	ResNet-50	0.5	50%	50%	72.4
ThiNet	ResNet-50	0.5	50%	50%	72.2
LCP(our)	ResNet-50	0.5	50%	50%	**73.3**
DCP	ResNet-50	0.7	70%	70%	70.2
ThiNet	ResNet-50	0.7	70%	70%	70.8
LCP(our)	ResNet-50	0.7	70%	70%	**71.7**

Table 3: The pruning results on MS COCO2017. The backbone is ResNet-50, We conduct channel pruning from conv2-x to conv4-x with compression ratio 0.7. Small, medium, large are the size of objects.

Method	small	medium	large	AP_{50}	AP_{75}	mAP
Original	4.2	22.5	39.0	37.3	22.7	21.9
DCP	2.8	17.2	33.0	31.8	17.8	17.8
LCP	**4.1**	20.4	38.2	35.5	**21.6**	**20.9**
Relative improv.%	**46.4**	18.6	15.8	11.6	**21.3**	**17.4**

Table 4: The pruning results on COCO. We conduct channel pruning from conv2-x to conv4-x.

Method	backbone	η	flops↓	params↓	mAP
Original	ResNet-50	0	0	0	21.9
DCP	ResNet-50	0.5	50%	50%	21.2
ThiNet	ResNet-50	0.5	50%	50%	22.6
LCP(our)	ResNet-50	0.5	50%	50%	**23.1**
DCP	ResNet-50	0.7	70%	70%	17.8
ThiNet	ResNet-50	0.7	70%	70%	20.2
LCP(our)	ResNet-50	0.7	70%	70%	**20.9**

From the results, our method achieves a better performance than the DCP and ThiNet, which further illustrates the effectiveness of our approach. It is noted that compared with DCP, LCP has larger gain on small objects. In addition, the higher the IoU threshold, the greater improvement of our method. This indicates that our method retains more localization information and can obtain more accurate predictions.

Ablation Analysis

Gradient Analysis. In this section, we prune the VGG-16 from conv1-1 to conv5-3 with compression ratio 0.75 On PASCAL VOC. Then we count the percentage of the gradients generated by the three losses during the pruning process. From Fig. 6, we see that the gradient of regression loss play a important role during the pruning process, which proves that the localization information is necessary. The gradient generated by reconstruction error only works in the shallow layers while the localization-aware loss contributes to the channel pruning process each layer.

Component Analysis. In this section, in order to verify the effectiveness of the two points we propose, we prune the SSD based on ResNet-50 by 70% with different combinations of our points. We report the results in Tab.5. From the results, we can get that each part of the method we propose contributes to the performance.

Loss Analysis. In order to explore the importance of the gradient of regression loss, we prune the SSD based on VGG-16 by 75% with different losses. We report the results in Tab. 6. From the results, we can know that the performance of our method drops a lot without the gradient of the regression loss during the pruning stage, which shows that the regres-

Figure 7: The predictions of original SSD, models pruned by Thinet and our LCP. We prune the VGG-16 by 75% on PASCAL VOC.

Table 5: The pruning results on PASCAL VOC2007. We conduct channel pruning from conv2-x to conv4-x. CR means Contextual ROIAlign.

Method	backbone	flops↓	params↓	mAP
DCP	ResNet-50	70%	70%	70.2
LCP+ROIAlign	ResNet-50	70%	70%	**71.1**
LCP+CR	ResNet-50	70%	70%	**71.7**

Table 6: The pruning results on PASCAL VOC2007. We conduct channel pruning from conv1-1 to conv5-3 .

Method	backbone	η	flops↓	params↓	mAP
Original	VGG-16	0	0	0	77.4
$\mathcal{L}_{re}$+$\mathcal{L}_{ac}$	VGG-16	0.75	75%	75%	74.7
$\mathcal{L}_{re}$+$\mathcal{L}_{ac}$+$\mathcal{L}_{ar}$	VGG-16	0.75	75%	75%	**75.2**

sion branch contains important localization information.

Visualization of predictions

In this section, we prune the SSD based on VGG-16 by 75% and we compare the original model with the pruned models. From Fig. 7, we can find that the predictions of our method are closed to the predictions of the original model while the predictions of ThiNet are far away. It is proved that our method reserve more localization information for bounding box regression.

Conclusions

In this paper, we propose a localization-aware auxiliary network which allows us to conduct channel pruning directly for object detection. First, we design the auxiliary network with a contextual roialign layer which can obtain precise localization information of the default boxes by pixel alignment and enlarges the receptive fields of the default boxes when pruning shallow layers. Then, we construct a loss function for object detection task which tends to keep the channels that contain the key information for classification and regression. Visualization shows our method reserves layers with more localization information. Moreover, extensive experiments demonstrate the effectiveness of our method.

References

[Alvarez and Salzmann 2016] Alvarez, J. M., and Salzmann, M. 2016. Learning the number of neurons in deep networks. In *Advances in Neural Information Processing Systems*, 2270–2278.

[Courbariaux et al. 2016] Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*.

[Everingham et al. 2010] Everingham, M.; Van Gool, L.; Williams, C. K.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision* 88(2):303–338.

[Guo, Yao, and Chen 2016] Guo, Y.; Yao, A.; and Chen, Y. 2016. Dynamic network surgery for efficient dnns. In *Advances In Neural Information Processing Systems*, 1379–1387.

[Han et al. 2015] Han, S.; Pool, J.; Tran, J.; and Dally, W. 2015. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, 1135–1143.

[Han, Mao, and Dally 2015] Han, S.; Mao, H.; and Dally, W. J. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

[Hassibi and Stork 1993] Hassibi, B., and Stork, D. G. 1993. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in neural information processing systems*, 164–171.

[He et al. 2016] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

[He, Zhang, and Sun 2017] He, Y.; Zhang, X.; and Sun, J. 2017. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 1389–1397.

[Hu et al. 2016] Hu, H.; Peng, R.; Tai, Y.-W.; and Tang, C.-K. 2016. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*.

[Jiang et al. 2018] Jiang, C.; Li, G.; Qian, C.; and Tang, K. 2018. Efficient dnn neuron pruning by minimizing layer-wise nonlinear reconstruction error. In *IJCAI*, volume 2018, 2–2.

[Jin et al. 2016] Jin, X.; Yuan, X.; Feng, J.; and Yan, S. 2016. Training skinny deep neural networks with iterative hard thresholding methods. *arXiv preprint arXiv:1607.05423*.

[Krizhevsky, Sutskever, and Hinton 2012] Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

[LeCun, Denker, and Solla 1990] LeCun, Y.; Denker, J. S.; and Solla, S. A. 1990. Optimal brain damage. In *Advances in neural information processing systems*, 598–605.

[Li et al. 2016] Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; and Graf, H. P. 2016. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*.

[Li, Zhang, and Liu 2016] Li, F.; Zhang, B.; and Liu, B. 2016. Ternary weight networks. *arXiv preprint arXiv:1605.04711*.

[Lin et al. 2014] Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755. Springer.

[Liu et al. 2016] Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*, 21–37. Springer.

[Liu et al. 2017] Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; and Zhang, C. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, 2736–2744.

[Luo, Wu, and Lin 2017] Luo, J.-H.; Wu, J.; and Lin, W. 2017. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, 5058–5066.

[Noh, Hong, and Han 2015] Noh, H.; Hong, S.; and Han, B. 2015. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, 1520–1528.

[Ren et al. 2015] Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

[Rezatofighi et al. 2019] Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; and Savarese, S. 2019. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 658–666.

[Russakovsky et al. 2015] Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision* 115(3):211–252.

[Simonyan and Zisserman 2014] Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[Wang et al. 2016] Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; and Van Gool, L. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, 20–36. Springer.

[Wen et al. 2016] Wen, W.; Wu, C.; Wang, Y.; Chen, Y.; and Li, H. 2016. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, 2074–2082.

[Zhou et al. 2017] Zhou, A.; Yao, A.; Guo, Y.; Xu, L.; and Chen, Y. 2017. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv preprint arXiv:1702.03044*.

[Zhuang et al. 2018] Zhuang, Z.; Tan, M.; Zhuang, B.; Liu, J.; Guo, Y.; Wu, Q.; Huang, J.; and Zhu, J. 2018.

Discrimination-aware channel pruning for deep neural networks. In *Advances in Neural Information Processing Systems*, 875–886.