Learning Efficient Convolutional Networks through Network Slimming

(2017 IEEE International Conference on Computer Vision)

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, Changshui Zhang

2021.08.26

세종대학교 무인이동체공학전공

곽수지

Contents

1. Introduction of Network Slimming
2. Advantages of channel-level sparsity
3. Scaling factors and Sparsity-induced penalty
4. Conclusion

Network Slimming

***** Channel-level sparsity in the network**

GOAL:

1) Reduce the model size
2) Decrease the run-time memory footprint
3) Lower the number of computing operations, without compromising accuracy

The proposed method directly applies to modern CNN architectures

- Introducing minimum overhead to the training process
- Requires no special SW/HW accelerators for the resulting models

Introduction

state-of-the-art CNN models including VGGNet, ResNet and DenseNet 에 적용

VGGNet:

a 20x reduction in model size and a 5x reduction in computing operations

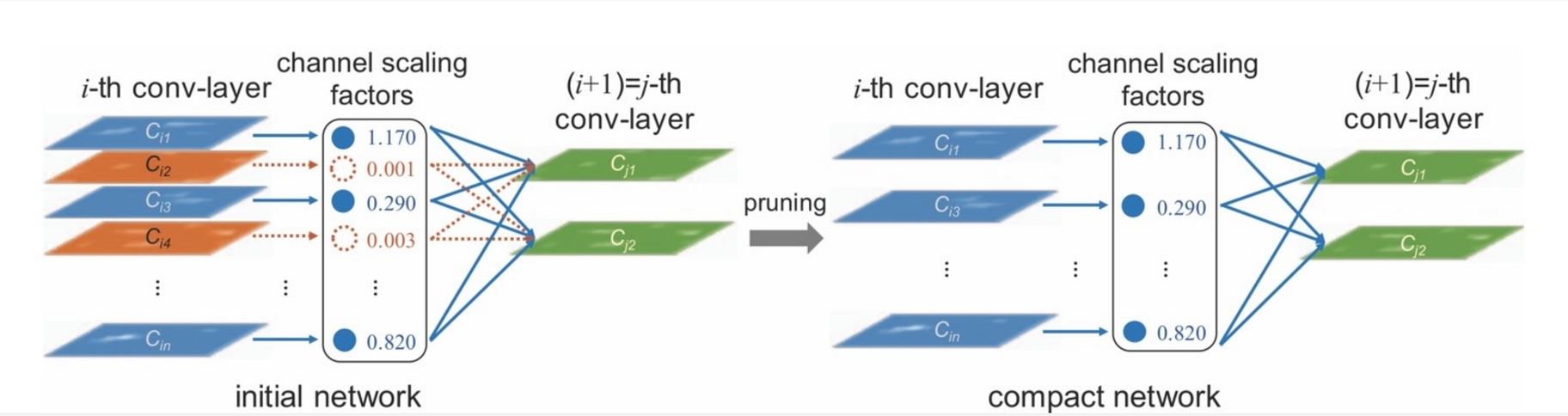
Related Work

Low-rank approximation, network quantization, binarization, weight pruning, dynamic inference

→ some techniques require specially designed sw/hw accelerators for execution speedup.

2. Sparsify the network → require special SW/HW accelerators to harvest the gain in memory or time savings

Introduction

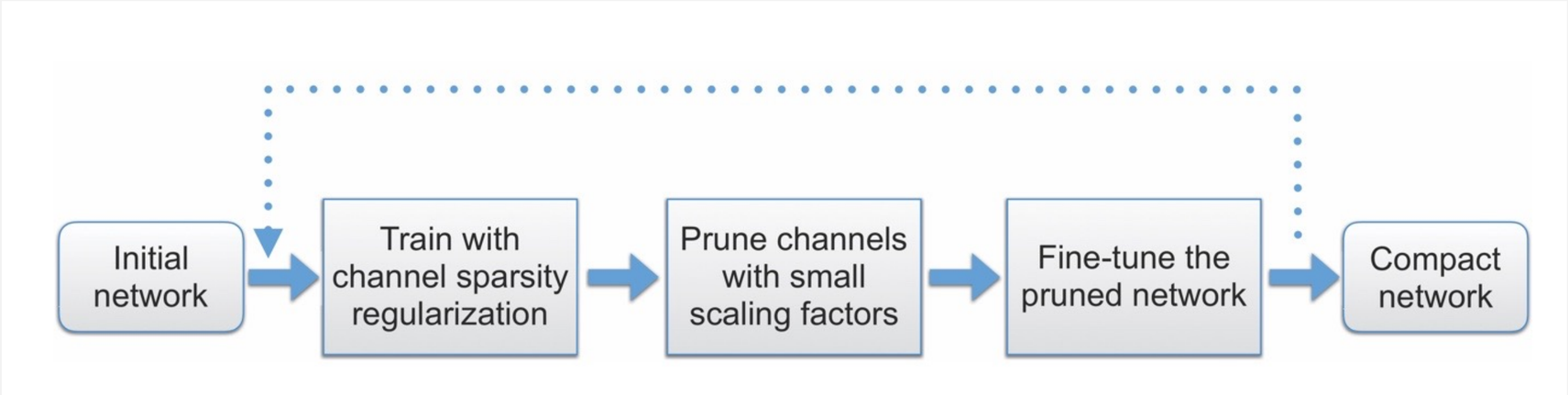


- Rarely hurt the performance
- Temporarily degrade the performance

  → can be compensated by the followed fine-tuning

: After pruning, the resulting narrower network is much more compact in terms of model size, runtime memory, and computing operations compared to the initial wide network.

Training



1. channel sparsity regularization이 추가된 Loss로 CNN 모델 학습
2. Scaling factors가 작은 것들에 대해 임계 값을 도입하여 가지치기
3. 일시적으로 하락한 성능, Fine-tuning으로 복구

→ **COMPACT NETWORK !!!**

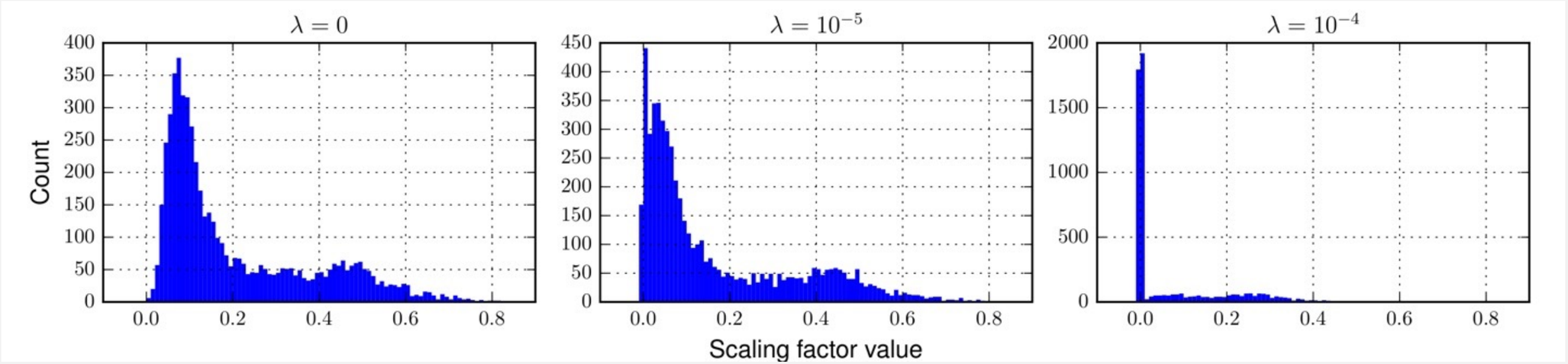
Introduction

L1 regularization on the scaling factors in batch normalization(BN) layers

기존 CNN 구조를 변경하지 않는 방법

→ enables us to identify insignificant channels

$$L = \sum_{(x,y)} l(f(x, W), y) + \lambda \sum_{\gamma \in \Gamma} g(\gamma)$$



Advantages of channel-level sparsity

Advantages:

1. Nice tradeoff between flexibility and ease of implementation

-It can be applied to any typical CNNs or fully connected networks

2. Resulting network is a “thinned” version of the unpruned network,

which can be efficiently inferenced on convetional CNN platforms.

Scaling Factors and Sparsity-induced Penalty

$$L = \sum_{(x,y)} l(f(x, W), y) + \lambda \sum_{\gamma \in \Gamma} g(\gamma)$$

$$\hat{z} = \frac{z_{in} - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}; \quad z_{out} = \gamma \hat{z} + \beta$$

Experiments (paper)

(a) Test Errors on CIFAR-10

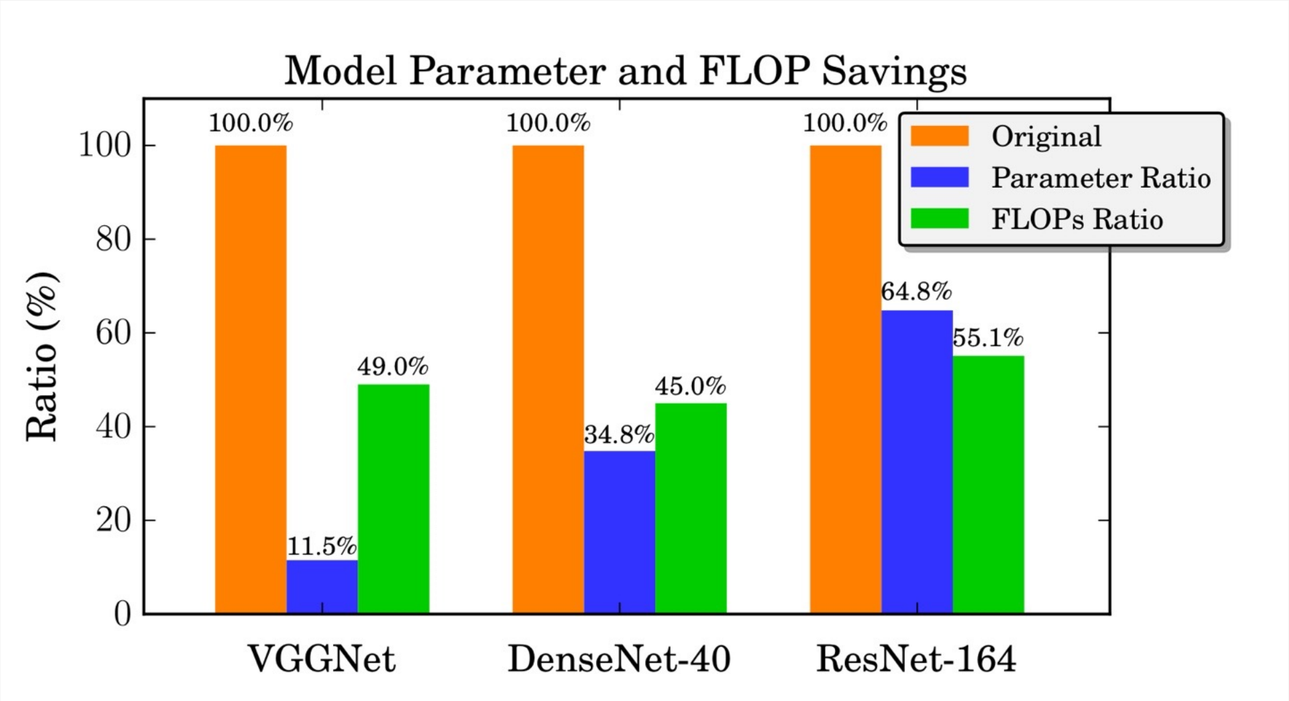
Model	Test error (%)	Parameters	Pruned	FLOPs	Pruned
VGGNet (Baseline)	6.34	20.04M	-	7.97×10^8	-
VGGNet (70% Pruned)	**6.20**	2.30M	88.5%	3.91×10^8	51.0%
DenseNet-40 (Baseline)	6.11	1.02M	-	5.33×10^8	-
DenseNet-40 (40% Pruned)	**5.19**	0.66M	35.7%	3.81×10^8	28.4%
DenseNet-40 (70% Pruned)	5.65	0.35M	65.2%	2.40×10^8	55.0%
ResNet-164 (Baseline)	5.42	1.70M	-	4.99×10^8	-
ResNet-164 (40% Pruned)	**5.08**	1.44M	14.9%	3.81×10^8	23.7%
ResNet-164 (60% Pruned)	5.27	1.10M	35.2%	2.75×10^8	44.9%

(b) Test Errors on CIFAR-100

Model	Test error (%)	Parameters	Pruned	FLOPs	Pruned
VGGNet (Baseline)	26.74	20.08M	-	7.97×10^8	-
VGGNet (50% Pruned)	**26.52**	5.00M	75.1%	5.01×10^8	37.1%
DenseNet-40 (Baseline)	25.36	1.06M	-	5.33×10^8	-
DenseNet-40 (40% Pruned)	**25.28**	0.66M	37.5%	3.71×10^8	30.3%
DenseNet-40 (60% Pruned)	25.72	0.46M	54.6%	2.81×10^8	47.1%
ResNet-164 (Baseline)	23.37	1.73M	-	5.00×10^8	-
ResNet-164 (40% Pruned)	**22.87**	1.46M	15.5%	3.33×10^8	33.3%
ResNet-164 (60% Pruned)	23.91	1.21M	29.7%	2.47×10^8	50.6%

(c) Test Errors on SVHN

Model	Test Error (%)	Parameters	Pruned	FLOPs	Pruned
VGGNet (Baseline)	2.17	20.04M	-	7.97×10^8	-
VGGNet (60% Pruned)	**2.06**	3.04M	84.8%	3.98×10^8	50.1%
DenseNet-40 (Baseline)	1.89	1.02M	-	5.33×10^8	-
DenseNet-40 (40% Pruned)	**1.79**	0.65M	36.3%	3.69×10^8	30.8%
DenseNet-40 (60% Pruned)	1.81	0.44M	56.6%	2.67×10^8	49.8%
ResNet-164 (Baseline)	**1.78**	1.70M	-	4.99×10^8	-
ResNet-164 (40% Pruned)	1.85	1.46M	14.5%	3.44×10^8	31.1%
ResNet-164 (60% Pruned)	1.81	1.12M	34.3%	2.25×10^8	54.9%



Experiments (paper)

VGG-A	Baseline	50% Pruned
Params	132.9M	23.2M
Params Pruned	-	82.5%
FLOPs	4.57×10^{10}	3.18×10^{10}
FLOPs Pruned	-	30.4%
Validation Error (%)	36.69	36.66

Table 2: Results on ImageNet.

Model	Test Error (%)	Params Pruned	#Neurons
Baseline	1.43	-	784-500-300-10
Pruned [35]	1.53	83.5%	434-174-78-10
Pruned (ours)	1.49	84.4%	784-100-60-10

Table 3: Results on MNIST.

(a) Multi-pass Scheme on CIFAR-10

Iter	Trained	Fine-tuned	Params Pruned	FLOPs Pruned
1	6.38	6.51	66.7%	38.6%
2	6.23	6.11	84.7%	52.7%
3	5.87	6.10	91.4%	63.1%
4	6.19	6.59	95.6%	77.2%
5	5.96	7.73	98.3%	88.7%
6	7.79	9.70	99.4%	95.7%

(b) Multi-pass Scheme on CIFAR-100

Iter	Trained	Fine-tuned	Params Pruned	FLOPs Pruned
1	27.72	26.52	59.1%	30.9%
2	26.03	26.52	79.2%	46.1%
3	26.49	29.08	89.8%	67.3%
4	28.17	30.59	95.3%	83.0%
5	30.04	36.35	98.3%	93.5%
6	35.91	46.73	99.4%	97.7%

Experiments (paper)

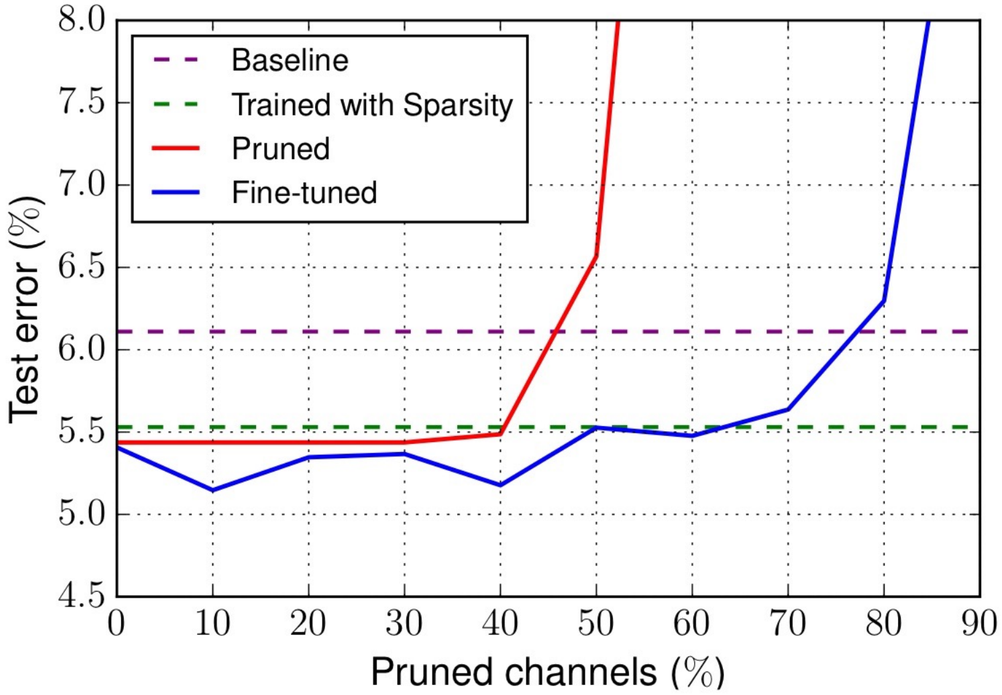
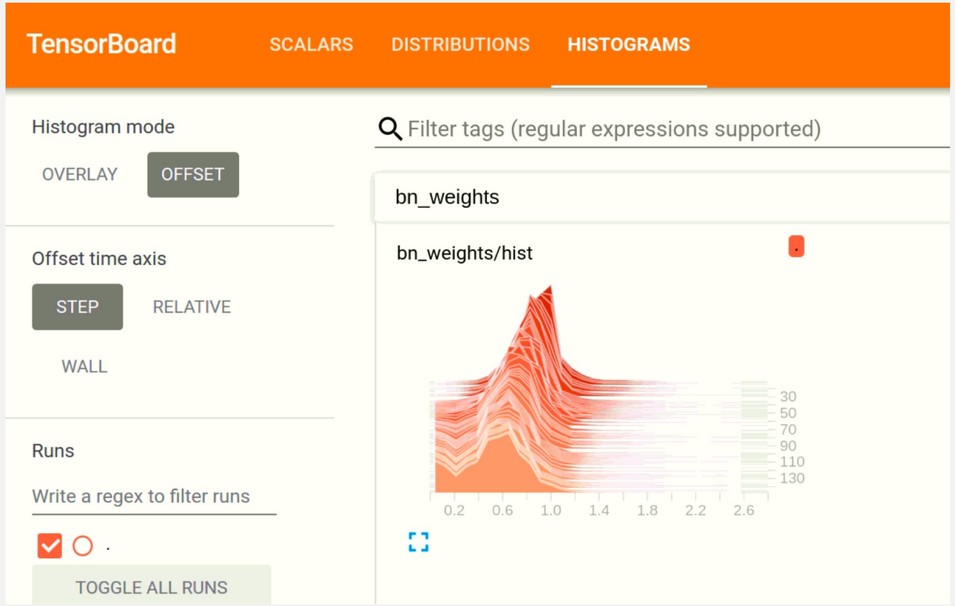


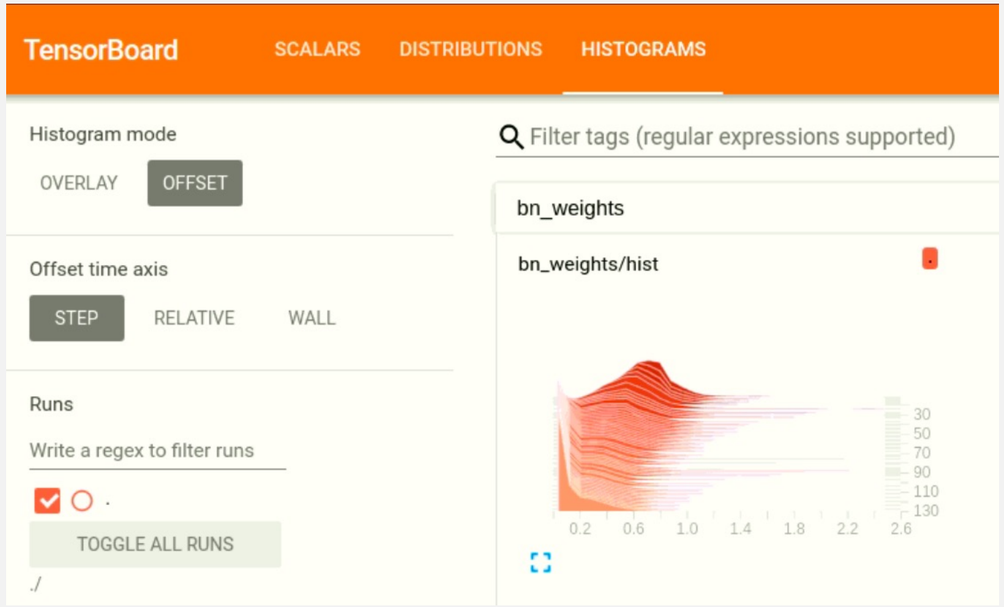
Figure 5: The effect of pruning varying percentages of channels, from DenseNet-40 trained on CIFAR-10 with λ=10^{-5}.

Experiments

Channel Pruning Result(-sr —s e-3)

Aa 이름	Before	Prune 0.5	Prune 0.6	Prune 0.7
mAP	0.809014	0.809379	0.651305	0.413506
Parameters	61529119	29813916	26197866	21464464
Inference	0.0099	0.0074	0.0070	0.0065





Conclusions

6. Conclusion

We proposed the network slimming technique to learn more compact CNNs. It directly imposes sparsity-induced regularization on the scaling factors in batch normalization layers, and unimportant channels can thus be automatically identified during training and then pruned. On multiple datasets, we have shown that the proposed method is able to significantly decrease the computational cost (up to 20×) of state-of-the-art networks, with no accuracy loss. More importantly, the proposed method simultaneously reduces the model size, run-time memory, computing operations while introducing minimum overhead to the training process, and the resulting models require no special libraries/hardware for efficient inference.

BN의 scaling factor에 sparsity-induced regularization을 적용하여 불필요한 채널 제거

정확도의 큰 손실 없이 모델 경량화 가능

Model size, run-time memory, computing operation 측면에서의 이득

Special libraries, SW/HW가 필요 없음