A SIMPLE HYBRID FILTER PRUNING FOR EFFICIENT EDGE INFERENCE

ID: 4045

S.H.Shabbeer Basha, Sheethal N Gowda, Jayachandra Dakala

DryvAmigo, Bangalore, India.

{shabbeer.sh,sheethal.ng,jayachandra.d}@dryvamigo.com

icassp 2022 Singapore

Abstract

Deployment of deep CNNs limited due to their computational and memory requirement. Many embedded friendly models such as MobileNet, ShuffleNet, SqueezeNet, and many more are proposed to serve this purpose. But these models are still not compact enough to deploy on edge devices. The popular metric based pruning methods (which are aimed at pruning insignificant and redundant filters) could achieve limited compression for embedded friendly models such as MobileNet. In this paper, we propose a novel hybrid filter pruning method that prunes both redundant and insignificant filters at the same time. Additionally, we have designed custom regularizers that enable us to prune more filters from convolutional layers. Pruning experiments are conducted on MobileNetv1 based Single-Shot Object Detector (SSD) for face detection problem. Through our experiments, we could prune 40.11% of parameters and reduce 67.03% of FLOPs from MobileNetv1 with a little drop in model performance (1.67 mAP on MS COCO). On an ARM based edge device, the inference time is reduced from 198ms to 84ms.

Introduction and Related Works

- Deep CNNs requires more space due to which their deployment is limited
- Neural Network Compression methods [1] aim to reduce memory and computational requirement.
- Broadly, these methods are classified into three types: i) Network pruning ii) Weight quantization [2] iii) Knowledge distillation [3].

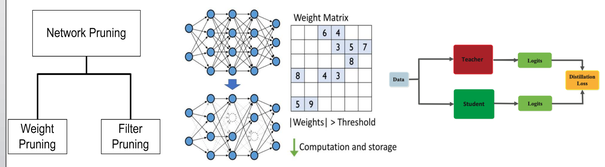


Fig. 1 a) Network pruning b) Weight pruning [4] c) Knowledge distillation [3]

- Filter pruning methods are generic, popular compared to other network compression methods.
- The existing magnitude based and redundant filter pruning methods mostly employed on CNNs like VGG-16, ResNet-50, etc.,.
- It is recommended to use embedded friendly models for real time applications such as Driver Monitoring System (DMS).
- Most of the network compression methods on embedded friendly models are based on knowledge distillation.

Contributions

1. Proposed a Hybrid Filter Pruning (HFP) method to prune filters that are insignificant and redundant.
2. Achieving higher model compression through finetuning the pruned models with custom regularizers.
3. Developed and compared the state-of-the-art magnitude based and redundant filter pruning methods with the proposed method.

Methodology

Preliminaries:

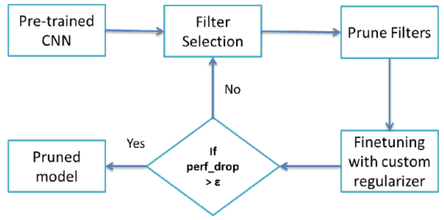
Magnitude based filter pruning methods

L-1 norm [5], L-2 norm [6]

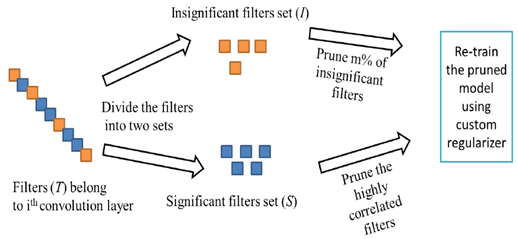
Redundant filter pruning methods [7]

Similarity measure (like Cosine similarity, Pearson's Correlation coefficient)

The proposed filter pruning framework



Hybrid Filter Pruning



Object Detection (SSD) [8]

$$L = \frac{1}{N}\left(L_{conf}(x,c) + \lambda_1 L_{loc}(x,l,g) + \lambda_2 \sum_{i=1}^{n} w_i^2\right)$$

Custom regularizers

$$C_1 = -\left(\sum_{\forall f_k \in R_i}^{M} \ell_1(f_k)\right)$$

$$C_2 = -\left(\sum_{\forall f_k \in R_i}^{M} \ell_2(f_k)\right)$$

i indicates either 1 or 2.

λ_2 is considered as 0.1.

Custom Loss

$$L = \frac{1}{N}\left(L_{conf}(x,c) + \lambda_1 L_{loc}(x,l,g) - \lambda_2 C_i\right)$$

Experiments

Training Details:

- MobileNetv1 based Single Shot Object Detection (SSD) model [8]
- Initially trained on MS COCO dataset [9]
- Finetuned on WiderFace dataset [10]
- Training images: MS COCO, WiderFace
- Validation images: MS COCO, WiderFace
- Test images: FDDB [11]

Results

Object Detection Results

S.No.	Method	mAP	Remaining Parameters (Drop %)	Remaining FLOPs (Drop %)
1	base model	25.63	6.83M (0)	1.23B (0)
2	Knowledge Distillation [11]	27.70	6.83M (0)	1.23B (0)
3	LFFSD [17]	20.40	6.50M (4.4)	-
4	ℓ_1-norm - I	25.22	6.45M (05.51)	1.11B (09.77)
	ℓ_1-norm - V	24.52	5.26M (22.91)	0.74B (39.91)
	ℓ_1-norm - VI	21.86	5.03M (26.35)	0.67B (45.52)
5	ℓ_1-norm + C_1 - I	24.66	6.45M (05.51)	1.11B (09.77)
	ℓ_1-norm + C_1 - V	24.21	5.26M (22.91)	0.74B (39.91)
	ℓ_1-norm + C_1 - VII	22.53	4.82M (29.42)	0.61B (50.40)
6	ℓ_2-norm - I	24.58	6.45M (05.51)	1.11B (09.77)
	ℓ_2-norm - V	24.24	5.26M (22.91)	0.74B (39.91)
	ℓ_2-norm - VII	22.60	4.66M (26.20)	0.54B (56.09)
7	ℓ_2-norm + C_2 - I	25.05	6.45M (05.51)	1.11B (09.77)
	ℓ_2-norm + C_2 - V	24.28	5.26M (22.91)	0.74B (39.91)
	ℓ_2-norm + C_2 - XI	23.82	4.68M (31.47)	0.52B (57.72)
8	CFP with Optimization [7] - I	23.84	6.81M (00.02)	1.21B (01.62)
	CFP with Optimization [7] - V	23.43	6.79M (00.05)	1.15B (06.50)
9	HFP - I	25.14	6.64M (02.78)	1.17B (04.87)
	HFP - V	25.08	5.97M (12.59)	0.96B (21.95)
	HFP - VII	23.00	5.30M (22.40)	0.75B (39.02)
10	ℓ_2-norm + C_2 - XV	24.64	4.45M (34.81)	0.49B (59.71)
	ℓ_2-norm + C_2 - XXII	24.79	4.18M (38.67)	0.43B (65.08)
	ℓ_2-norm + C_2 - XXV	23.96	**4.09M (40.11)**	**0.40B (67.03)**
11	HFP + C_2 - I	25.49	6.64M (02.78)	1.17B (04.87)
	HFP + C_2 - V	28.00	5.97M (12.59)	0.96B (21.95)
	HFP + C_2 - XI	**28.58**	5.18M (24.15)	0.72B (41.46)
	HFP + C_2 - XVIII	24.49	4.50M (34.11)	0.52B (57.72)

Face Detection Results

S.No	Pruning Method	recall (#FPs=500)	Inference Time (in ms)	Memory required (in MB)
1	base model	91.85	198	22
2	ℓ_1-norm - V	91.28	127	16
3	ℓ_1-norm + C_1 - V	91.08	127	16
4	ℓ_2-norm - V	90.96	127	16
5	ℓ_2-norm + C_2 - V	91.33	127	16
6	CFP with optimization [7] - V	90.36	171	21
7	HFP - V	91.04	154	19
8	ℓ_2-norm + C_2 - XV	89.40	96	13
9	ℓ_2-norm + C_2 - XXV	88.78	84	12
10	HFP + C_2 - XI	90.98	121	15
11	HFP + C_2 - XVIII	89.97	96	13

Face recognition (InceptionResNetv1) Results

S.No.	Method	validation accuracy	Remaining Parameters (Drop %)	Remaining FLOPs (Drop %)
1	Base model	97.73	23.4M (0)	1.41B (0)
2	ℓ_1-norm - I	98.16	20.8M (10.95)	1.24B (12.27)
	ℓ_1-norm - V	98.2	12.9M (44.58)	0.72B (49.21)
	ℓ_1-norm - XII	96.3	5.44M (76.77)	0.25B (82.07)
3	ℓ_1-norm + C_1 - I	98.33	20.8M (10.95)	1.24B (12.27)
	ℓ_1-norm + C_1 - V	98.733	12.9M (44.58)	0.72B (49.21)
	ℓ_1-norm + C_1 - XII	96.63	5.44M (76.77)	0.25B (82.07)
4	ℓ_2-norm - I	98.33	20.8M (10.95)	1.24B (12.27)
	ℓ_2-norm - V	98.26	12.9M (44.58)	0.72B (49.21)
	ℓ_2-norm - X	97.03	7.03M (69.98)	0.34B (75.39)
5	ℓ_2-norm + C_2 - I	98.33	20.8M (10.95)	1.24B (12.27)
	ℓ_2-norm + C_2 - V	97.80	12.9M (44.58)	0.72B (49.21)
	ℓ_2-norm + C_2 - X	98.23	7.03M (69.98)	0.34B (75.39)
6	HFP - I	98.40	21.1M (09.52)	1.27B (09.85)
	HFP - II	98.53	19.2M (18.03)	1.15B (18.75)
	HFP - III	98.26	17.3M (25.98)	1.04B (26.47)

Conclusions

- Employing the existing filter pruning methods on embedded friendly models results in significant performance degradation.
- The proposed Hybrid Filter Pruning method enables the CNN to consists the filters that are significant and unique.
- The proposed regularizers enable us to achieve higher compression.
- In the process of pruning, we obtained compact models that improves the performance by 3% compared to base model.

Bibliography

[1]. Neill, James O. "An overview of neural network compression." *arXiv preprint arXiv:2006.03669* (2020).

[2]. Gholami, Amir, et al. "A survey of quantization methods for efficient neural network inference." *arXiv preprint arXiv:2103.13630* (2021).

[3]. Wang, Lin, and Kuk-Jin Yoon. "Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).

[4]. Han, Song, et al. "Learning both weights and connections for efficient neural network." *Advances in neural information processing systems* 28 (2015).

[5]. Li, Hao, et al. "Pruning filters for efficient convnets." *arXiv preprint arXiv:1608.08710* (2016).

[6]. He, Yang, et al. "Soft filter pruning for accelerating deep convolutional neural networks." *arXiv preprint arXiv:1808.06866* (2018).

[7]. Singh, Pravendra, et al. "Leveraging filter correlations for deep model compression." *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020.

[8]. Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg, "Ssd: Single shot multibox detector," in European conference on computer vision. Springer, 2016.

[9]. Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll´ar, and C Lawrence Zitnick, "Microsoft coco: Common objects in context," in European conference on computer vision. Springer, 2014, pp. 740–755.

[10]. Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang, "Wider face: A face detection benchmark," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[11]. Vidit Jain and Erik Learned-Miller, "Fddb: A benchmark for face detection in unconstrained settings," Tech. Rep., UMass Amherst technical report, 2010.

printed by MegaPrint Inc. www.postersession.com