Performance-Enhancing Network Pruning for Crowd Counting

Lei Liu[a,b], Saeed Amirgholipour[b], Jie Jiang[a,*], Wenjing Jia[b], Michelle Zeibots[c], Xiangjian He[b,*]

[a]School of Instrumentation Science and Opto-Electronics Engineering, Beihang University, China
[b]Global Big Data Technologies Centre, University of Technology Sydney, Australia
[c]Institute for Sustainable Futures, University of Technology Sydney, Australia

Abstract

The Counting Convolutional Neural Network (CCNN) has been widely used for crowd counting. However, they typically end up with a complicated network model resulting in a challenge for real-time processing. Existing solutions aim to reduce the size of the network model, but unavoidably sacrifice the network accuracy. Different from existing pruning solutions, in this paper, a new pruning strategy is proposed by considering the contributions of various filters to the final result. The filters in the original CCNN model are grouped into positive, negative and irrelevant types. We prune the irrelevant filters of which feature maps contain little information, and the negative filters determined by a mask learned from the training dataset. Our solution improves the results of the counting model without fine-tuning or retraining the pruned model. We demonstrate the advantages of our proposed approach on the problem of crowd counting. Our experimental results on benchmark datasets show that the network model pruned using our approach not only reduces the network size but also improves the counting accuracies by 4% to 17% less MAE than the state of the arts.

[*]Corresponding author

Email addresses: BY1417114@buaa.com.cn (Lei Liu), Saeed.AmirgholipourKasmani@student.uts.edu.au (Saeed Amirgholipour), jiangjie@buaa.edu.cn (Jie Jiang), Wenjing.Jia@uts.edu.au (Wenjing Jia), Michelle.E.Zeibots@uts.edu.au (Michelle Zeibots), Xiangjian.He@uts.edu.au (Xiangjian He)

Keywords: Model pruning, crowd counting, Convolutional Neural Networks

1. Introduction

Vision-based density estimation for accurately counting or estimating the number of people (or objects) in crowded scenes is a desirable technique in many real world applications including visual surveillance, traffic monitoring and crowd analysis. This is true especially in restricted, public places such as train stations, where incidents, traffic delay and even terrible stampedes have been reported due to overcrowding in these places. There is an urgent demand for real-time decision-making corresponding to crowd control and planning. Various real-world situations, such as occlusions, size and shape variations of people, and perspective distortion, have posed great challenges for practical solutions capable of handling such situations. Thus, correctly counting in crowded scenes has become an open and popular research problem nowadays [1].

The existing approaches for crowd counting can be roughly grouped into detection-based and feature-regression-based approaches. The detection-based approaches employ object detectors to detect or localize each person in the scene, and the counting is simply the number of total detections. These approaches can surpass human's performance in images with relatively large people sizes and sparse crowd densities [2, 3, 4]. However, in complex scenes with serious occlusions and extremely crowded scenes, detection-based approaches often fail to detect individuals and hence produce inaccurate counting [5]. The feature-regression-based approaches, e.g., [6, 7, 8, 9], on the other hand, aim to obtain the density function of an image containing people and then calculate the total count by integrating the densities over the whole image space. They have demonstrated a countable solution for handling highly crowded scenes.

Recently, a Counting Convolutional Neural Network (CCNN) model [6] has been proposed, which can learn to count people and produce density maps in images. Compared with the traditional hand-crafted feature based approaches, this approach has achieved much better accuracies in wider, real-world crowded scenes. However, high capacity deep networks typically have significant inference costs especially when being used in complex scenes. This has resulted in a challenge for embedded sensors or mobile devices, where computational and power resources are often very limited. Many research works have been reported to reduce the storage and computation costs of deep neural networks for various applications. A typical solution is to prune the weights with small magnitudes and then retrain the network aiming not to downgrade the overall accuracy significantly [10, 11, 12, 13]. Yet, to our best of knowledge, no one has attempted to simplify the deep network models in a way that also improves their accuracies.

In this paper, aiming to learn a lighter and more accurate deep network model, we propose a new strategy to prune the CCNN network [6] to not only simplify the network but also improve its accuracy. We examine the contributions of various filters in CCNN to the classification, and group the filters into positive, negative and irrelevant filters, respectively. Based on the feature maps of filters, we prune the irrelevant and negative filters so as to make the model lighter. Different from the existing pruning algorithms, our goal is to not only reduce the size of the model, but also improve the performance through our proposed pruning strategy. When tested on benchmark datasets, our solution not only prunes the network but also improves the accuracy by removing non-contributing and negatively contributing filters.

The main contributions of our work are summarized as follows.

- We propose a new pruning strategy that not only prunes the network but also improves the accuracy without fine tuning.
- We propose a simple but effective mechanism to prune the irrelevant filters based on the feature maps which have little information, as well as the negative filters learned from training data.

The rest of the paper is organized as follows. Section 2 shows the related work. In Section 3, the details about our proposed network pruning technique are given. The experiments conducted on various datasets are presented in Section 4. Finally, the paper concludes in Section 5.

2. Related work

Since detection-based counting approaches cannot be adapted to highly congested scenes, researchers try to deploy regression-based approaches to learn the relations between cropped image patches and their densities, and then calculate the number of particular objects. In recent years, many researchers [1] have developed deep learning models for image segmentation, classification and recognition, and have achieved very good results. Inspired by these, Convolutional Neural Network (CNN) models have been proposed to learn to count people and produce density maps in images simultaneously. These models work well for objects of a similar size in an image or a video. Sindagi and Patel [5] proposed an end-to-end cascaded network of CNNs that can learn globally relevant and discriminative features to estimate highly refined density maps with low counting errors. Onoro-Rubio and Lopez-Sastre in [6] proposed a regression model called Counting CNN (CCNN) and the Hydra CNN for multi-scaled crowd counting. The CCNN and Hydra CNN can map the appearance features of input image patches to corresponding density maps.

Inspired by the Hydra CNN model, some researchers have tried to utilize more complex deep models to solve the problem caused by the significant variance of people's appearance in a captured image/video. Deepak et al. [7] proposed a switching CNN to select the best CNN regressor for each of the different receptive fields and achieved better results. Kumagai1 et al. [8] proposed a mixture of CCNNs and adaptively selected multiple CNNs according to the appearance of a testing image for predicting the number of people. Zhang et al. [9] proposed a multi-column network from three independent CNNs, and then used the combined features of these three networks to get a density map. Li et al. [14] proposed CSRNet by combining VGG-16 and dilated convolution layers to aggregate multi-scale contextual information. All of these works have suggested some effective solutions for counting people in complex real-world senses. However, all of these models require very high computation resources for running, creating a challenge for embedded or mobile systems to adopt these models. Therefore, it makes sense to reduce the network complexity.

Network pruning and sharing have been adopted to reduce the network complexity and address the over-fitting issue. A recent trend in this direction is to prune redundant or non-informative weights in a pre-trained CNN model. For example, Srinivas and Babu [15] explored the redundancies among neurons, and proposed a data-free pruning method to remove redundant neurons. Pavlo Molchanov et al. [11] proposed a new method to prune filters in neural networks. Li et al. [12] proposed to prune the filters that have little effect on the accuracy. The deep compression method in [13] removed the redundant connections and quantized the weights, and then used Huffman coding to encode the quantized weights. In [16], a simple regularization method based on soft weight-sharing was proposed, and it included both quantization and pruning in one simple procedure. It is worthy to note that the above pruning schemes typically produce connection pruning in CNNs. However, all of these solutions achieve the pruning goal at the cost of losing accuracy to some extent.

For many cases, the networks may not have to be so complicated, so their complexity can be reduced. Then, is there any way to prune networks without decreasing their accuracies but with improved accuracies? It has been widely known that some filters contain little information for the final classification. However, according to our observation, some filters actually have negative impacts on the final classification. Therefore, pruning these filters will not only simplify the network models but also improve the network performance. In this paper, we propose a pruning approach and demonstrate its superiority on the application of crowd counting.

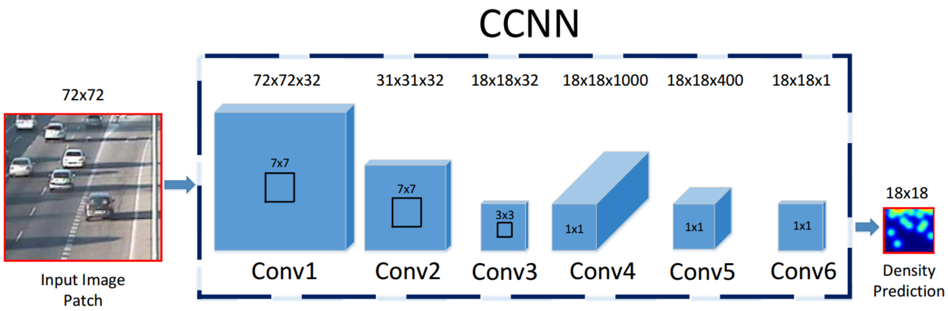

Figure 1: The structure of the CCNN model [6]

3. Network Pruning

Our work presented in this paper is initially designed for pruning the CCNN model [6] and can be applied to prune other crowd-counting network models. In this section, we first briefly introduce the CCNN-based crowd counting approach [6] and then present the details of our proposed pruning strategy.

3.1. CCNN

The CCNN approach [6] is formulated as a regression model that produces objects' density maps based on the corresponding appearances of image patches. Utilizing the sliding window technique, small patches are extracted from the input image as input to a pre-trained CNN model, which then produces an estimated density map for the corresponding image patch.

Fig. 1 shows the structure of the CCNN model, where input image patches are fed into a deep network to estimate their density maps.

Given an annotated training image I, where each of the targets is annotated with a dot (see Fig. 3), the density map, denoted as D_I, of the image, is defined as a sum of Gaussian functions centered at each dotted annotation as:

$$D_{\mathrm{I}}(p) = \sum_{\mu \in A_I} N(p; \mu, \Sigma), \tag{1}$$

where A_I is the set of dotted annotation of the image I, and $N(p; \mu, \Sigma)$ represents the evaluation of a normalized 2D Gaussian function, with a mean of μ and isotropic covariance matrix of Σ, evaluated at pixel p.

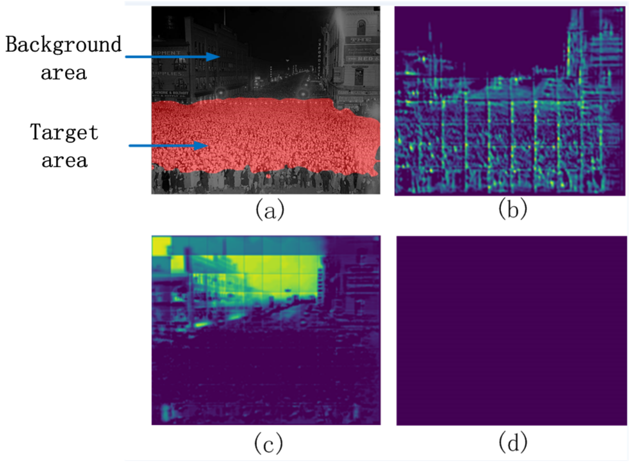

Figure 2: Example of the feature maps in layer conv4. (a) Input image. (b) Filters activating mostly on targets. (c) Filters activating mostly on background. (d) Filters with nearly no activations.

With the resultant density map D_I, the total object count N_I can be obtained by integrating the density map values D_I over the entire image space, as:

$$N_{\mathrm{I}} = \sum_{p \in I} D_{\mathrm{I}}(p). \tag{2}$$

Note that all the Gaussian functions are summed and normalized, so the total object count is preserved even when there is overlapping between targets.

3.2. Determining the types of filters

In training the CCNN model, the whole image is fed into the model. In crowd counting datasets, such as UCF and UCSD datasets, all of the images in training and testing datasets contain the target area, where the crowd is distributed, and the background area, where there are no people. According to [17], different filters activate on different targets of the images. Fig. 2 shows the activations of different filters in the feature maps corresponding to background and target areas, respectively.

In this figure, we can see that some feature maps have stronger activations on target area (see Fig. 2(b)), some filters activate mostly on background area (see

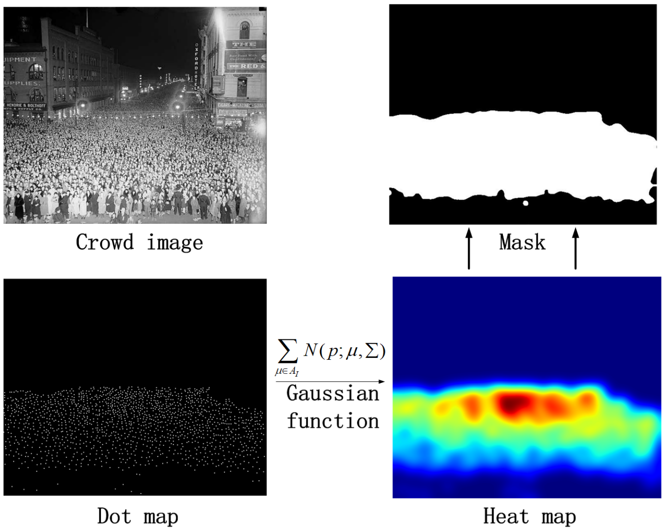

Figure 3: Learning the mask from an annotated training image.

Fig. 2(c)), and some feature maps contain nearly no activation (see Fig. 2(d)) and hence have little contribution to the classification result. Therefore, we can prune the model according to the activations of feature maps at different areas.

To examine the activations of feature maps corresponding to different areas, we learn a mask from annotated training images to identify the target area. Then, we define a simple mechanism to determine whether a filter makes positive or negative contributions to the classification, based on whether it mostly activates on target area or background area.

In a density map D_I, an intensity value larger than zero indicates that it has a non-zero density at the corresponding location. Thus, a binary mask, denoted by $M(x, y)$ (where (x, y) is the coordinates of the pixel p), corresponding to a target (when its value is 1) and background pixel (when its value is 0), respectively, can be derived from the density map function D_I as:

$$M(x,y) = \begin{cases} 1, & \text{if } D_I(p) > 0; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Fig. 3 shows an example of the areas derived from the mask. In Fig. 3, the white area corresponds to the crowd area, and the black area represents the

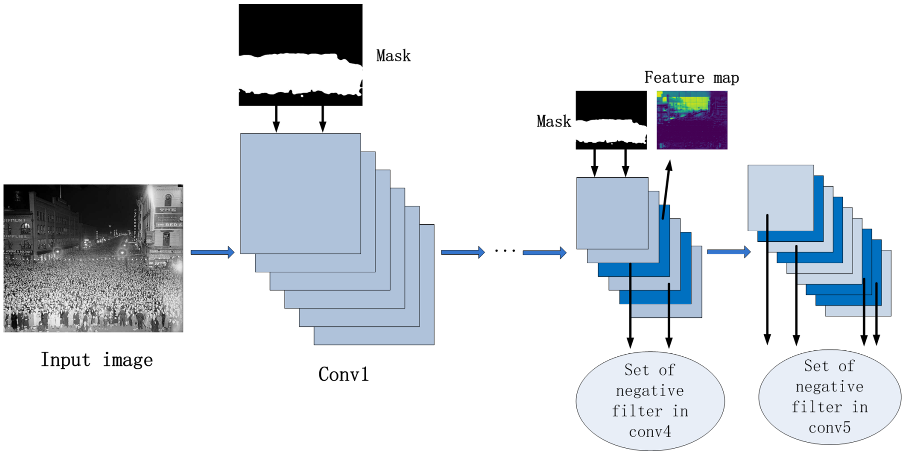
Mask
Feature map
Mask
Input image
Conv1
Set of negative filter in conv4
Set of negative filter in conv5

Figure 4: Model pruning with one mask

background. With the target and background learned using the mask, we can then easily determine whether a filter makes positive or negative contributions .

As shown in Fig. 4, after images are fed into the model, we apply the mask to all the feature maps in each layer. If the average magnitude of the background area (with the mask values equal to 0) divided by the average magnitude of the target area (with the mask values equal to 1) is higher than a pre-defined threshold η (selected based on experiments), it is concluded that the corresponding filter activates more on the background than the target area (see Fig. 2(c)) and it is defined as a possible negative filter.

3.3. Pruning filters and feature maps

Each training image has its own mask identifying its foreground and background, so it determines its own set of possible negative filters. In order to select the negative filters that are applicable to the entire dataset, we propose a simple voting mechanism to determine a maximum set of negative filters for the whole set of data. If a possible negative filter is included in most possible negative filter sets of training data, this filter will mostly likely be a negative filter for all data. Therefore, in this paper, a filter is pruned if it is included in

Table 1: The MAE results obtained on all sub-datasets in the UCF dataset obtained using the CCNN models with and without pruning.

	data0	data1	data2	data3	data4	MAE
CCNN [6]	775	476	510	276	373	488
Pruned CCNN	**759**	**396**	**488**	**247**	**335**	**445**

more than half of the negative filter sets.

To better illustrate this process, we take images from the UCF dataset. The UCF CC 50 dataset [11] consists of 50 pictures, collected from publicly available web images. Images in the UCF dataset are randomly split into five subsets and a 5-fold cross-validation is performed by following the standard setting in [18].

We randomly take a 10-image set from the training set and then create their masks from their dotted annotation maps. Then, the resultant mask is applied to the corresponding training images. If filters activate on more than half (*i.e.*, 5 in this example) of the training images, the filter is determined to be a possible negative filter according to Sect. 3.2 and will be pruned; If the feature map contains nearly no information, this filter is determined to be an irrelevant filter and will also be pruned.

Table 1 shows the MAE results obtained on all sub-datasets in the UCF dataset obtained using the CCNN models with and without pruning. As can be seen from this table that, after pruning, the accuracies are improved with the MAE reduced from 488 to 445. More comprehensive experiments are presented in Sect. 4.

3.4. Pruning of different layers

For the CNN model, in the shallow layers, the filters extract basic features, such as edges, anchors and so on. While in the deep layers, the filters tend to extract high level features, such as those to identify heads and bodies [17]. Therefore, we do not prune the shallow layers of the model, and only prune deep layers. What is more, we also prune those filters without activations shown on

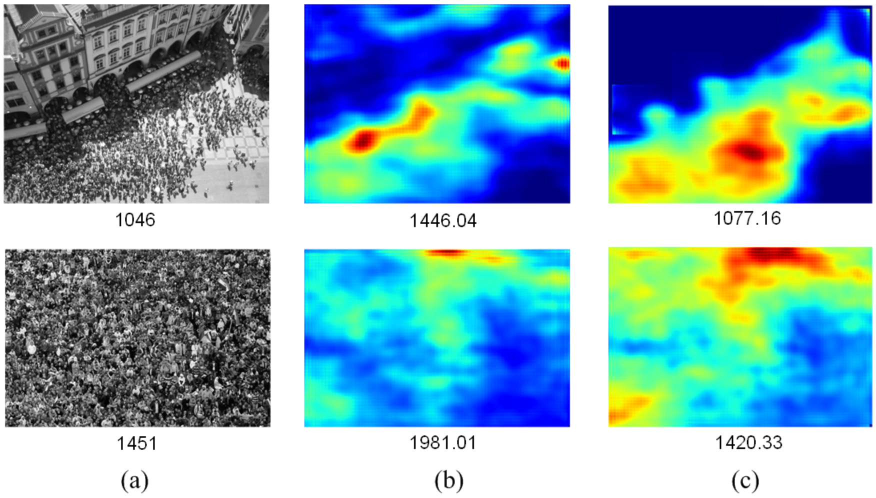

Figure 5: Examples of the density heat maps obtained with the original CCNN approach and our pruned CCNN model, where ground truth counts and estimation counts are shown underneath the images. (a) Input crowd images. (b) Density heat map obtained with the original CCNN. (c) Density heat map obtained with our pruned CCNN

the feature maps at all layers.

Fig. 6 shows the MAE results obtained for all five subsets of the UCF dataset when all of he single layers are pruned without re-tuning. In this figure, the red line in each graph is the MAE of the original CNN model for each sub-dataset. As shown in this figure, pruning deep layers, *e.g.*, pruning Conv5 layer (shown as the purple bars in the chart) vs pruning Conv2 layer (shown as the blue bars in the chart), tends to have more impact to the performance of the overall model. On the contrary, pruning shallow layers (*e.g.*, Conv2 or Conv3) always has little effects on the performance. We can make a conclusion that by pruning filters on deep layers, the counting results always get better. Thus, in our strategy, pruning is mostly carried out for Conv5 layer. Irrelevant filters containing little information in feature maps are pruned in Conv2 and Conv3, while in deep layers (Conv4 and Conv5), both irrelevant and negative filter are pruned.

4. Experiments

In this paper, we evaluate and compare our proposed pruning mechanism on crowd counting CCNN networks on four widely used benchmark datasets, *i.e.*, the UCF [19], UCSD [20], ShanghaiTech [21] datasets, and the TRANCOS dataset [22]. We implement our filter pruning algorithm based on the Caffe deep learning framework. When filters are pruned, a new model with fewer filters is created and the remaining parameters of the modified layers as well as the unaffected layers are copied into the new model.

4.1. Comparison with the original CCNN model

Fig. 5 shows two estimated density heat maps and counts obtained with the original CCNN and our pruned CCNN on two exemplar crowd images. As it can be seen, the estimation obtained with our pruned CCNN is much more accurate.

Next, following the convention of the similar works [7, 23, 8, 9] for crowd counting, we evaluate the performance of different approaches quantitatively on two datasets using the Mean Absolute Error (MAE), which is defined as:

$$MAE = 1/N \sum_{1}^{N} |z_i - z_i'|, \tag{4}$$

where N is the number of test images, z_i is the actual number of people in the i-th image, and z_i' is the estimated number of people in the i-th image. Roughly speaking, the lower the MAE is, the better accuracy the estimation method has.

4.1.1. Experimental results on the UCSD dataset

The UCSD dataset [20]contains 2,000 frames of video captured with a surveillance camera from a single scene on the UCSD campus. It has four subsets, namely 'maximal', 'downscale', 'upscale', and 'minimal' sub-datasets. The dataset provides the Region of Interest (ROIs) for each video frame. We use the ROI as the mask to determine the type of filters. As the scene of UCSD is fixed, we use one mask and follow the rules in the UCF to prune the model. The

Table 2: Comparison of the MAE results on the UCSD dataset

	maximal	downscale	upscale	minimal
CCNN [6]	1.70	1.79	1.13	1.50
Our pruned CCNN	**1.63**	**1.70**	**0.96**	**1.49**
Pruned Model size/MB	1.5	1.3	1.3	1.5

Table 3: Comparison of the MAE results TRANCOS datasets

Method	GAME0	GAME1	GAME2	GAME3
CCNN	12.49	16.58	20.02	22.41
Pruned CCNN	**11.25**	**14.26**	**16.43**	**19.72**
Pruned Model size/MB	1.7	1.1	2.1	1.8

results are shown in Table 2. Note that the size of the original model is 2.3MB. As shown in this table, after the pruning, the sizes of the models for the four sub-datasets are $1.5MB$, $1.3MB$, $1.3MB$ and $1.5MB$, respectively, decreased by 35% to 44%. Moreover, the accuracy obtained on all four sub-datasets are improved to some extents with reduced MAEs.

4.1.2. Experimental results on the TRANCOS dataset

TRANCOS [22] is a publicly available dataset, which provides a collection of 1,244 images of different traffic scenes, obtained from real video surveillance cameras, with a total of 46,796 annotated vehicles. The objects have been manually annotated using dots.

Table 3 reports the MAE results obtained on this dataset with the original CCNN model and our pruned model. As it can be seen from this table, the crowd count using the pruned CCNN is significantly higher than that of the original CCNN.

Table 4: Comparison of the proposed algorithm and other pruning algorithms on CCNN

Model/DATA	CCNN	[12]	ThiNet [24]	Distillation [25]	Our algorithm
UCSD maximal	1.70	1.73	1.72	1.72	**1.63**
UCSD minimal	1.50	1.50	1.51	1.55	**1.49**
UCSD upscale	1.13	1.14	1.14	1.11	**0.96**
UCSD downscale	1.79	1.78	1.81	1.84	**1.70**
UCF data0	775	775	782	768	**759**
UCF data1	476	476	450	483	**396**
UCF data2	510	510	515	529	**488**
UCF data3	276	276	279	293	**247**
UCF data4	373	373	377	364	**335**

4.2. Comparison with other pruning algorithms

We compare our proposed algorithm with other pruning algorithms applied to the CCNN model, *i.e.*, [12] and [24]. As the UCSD and UCF are the only two datasets for crowd counting based on CCNNs, we demonstrate the comparison on these two datasets.

Table 4 reports the MAE performance obtained using the proposed pruning algorithm and the algorithms proposed in [12] and [24]. As shown in this table, the MAE of our pruned CCNN on both datasets UCSD and UCF are significantly better than those of the original CCNN and the pruned CCNN with other pruning algorithms. Note that the results of [12], ThiNet [24] and Knowledge Distillation [25] are similar to those of the original CCNN and hence almost do not show any significant improvement on accuracy. However, our proposed pruning algorithm can not only reduce the size of the model, but also improve the results of the original CCNN model.

4.3. Pruning results on other crowd counting models

To demonstrate that our pruning method can also work with other models, we use the same method to prune other crowd counting models, *i.e.*, the MCNN model [21] and Switch-CNN models [7], on the ShanghaiTech dataset [21], UCF dataset [11] and UCSD dataset respectively. Note that, different from other pruning algorithms, we do not fine-tune or re-train our new model after pruning, but it still produces better results.

The ShanghaiTech dataset is a new large-scale crowd counting dataset including 482 images for congested scenes with 241,667 annotated persons. The results are shown in Tables 5 and 6.

Note that, in order to compare with MCNN and Switch-CNN, we add another metric, *i.e.*, Mean Squared Error (MSE), which is defined by:

$$MSE = \sqrt{\frac{1}{N}|\sum_{i=1}^{N}|C_i - C_i^{GT}|^2}, \tag{5}$$

where N is the number of images in the test set and C_i^{GT} is the ground truth of number of people in the test image, and C_i is the number of estimated counting.

As shown in these two tables, both of the MAE and MSE results obtained with the pruned MCNN and pruned Switch-CNN are significantly better than with the original MCNN and Switch-CNN, respectively. Moreover, the sizes of the original MCNN and Switch-CNN are 515KB, while the size of pruned models are decreased at different degree. This demonstrates that our algorithm can not only work on CCNN, but also on other counting models.

4.4. Impact of η

Moreover, we use a ratio learned from the dataset to determine the contribution of the filters in order to optimize the pruning effectiveness. In our work, the η is determined statistically through experiments. Fig. 6 shows the MAE results obtained on each of the five subset of the UCF dataset with different ratio η.

According to Fig. 6, the performance of different sub-models on this dataset achieves the best when using a ratio 2 to prune the Conv5 layer.

5. Conclusion

Table 5: Comparison of the results obtained on the ShanghaiTech dataset using the MCNN and Switch-CNN counting models with and without applying our pruning method

	partA		partB	
Method	MAE	MSE	MAE	MSE
MCNN [21]	110.2	173.2	26.4	41.3
Pruned MCNN	**100.5**	**170.5**	**23.5**	**39.7**
Switch-CNN[7]	90.4	135.0	21.6	30.1
Pruned Switch-CNN	**89.5**	**136.2**	**21.5**	**32.3**
Pruned MCNN size/KB	425		462	
Pruned Switch-CNN size/KB	436		477	

Table 6: Comparison of the results obtained on the UCF and UCSD datasets using the MCNN and Switch-CNN counting models with and without applying our pruning method

	UCF		UCSD	
Method	MAE	MSE	MAE	MSE
MCNN [21]	377.6	509.1	1.07	1.35
Pruned MCNN	**326.5**	**472.3**	**1.02**	**1.31**
Switch-CNN[7]	318.1	439.2	1.62	2.10
Pruned Switch-CNN	**305.1**	**410.9**	**1.44**	**1.73**
Pruned MCNN size/KB	413		389	
Pruned Switch-CNN size/KB	503		495	

In this paper, we have proposed a new pruning strategy for crowd counting that works with CCNN and other crowd counting models. Through identifying positive, negative and irrelevant filters according to the activations of feature maps, our solution has not only reduced the network size but also improved the accuracy by removing non-contributing and negatively contributing filters. Experimental results on benchmark datasets have shown that, compared with other existing pruning algorithms, our proposed technique can improve the accuracy of counting models without fine-tuning or retraining the pruned model,

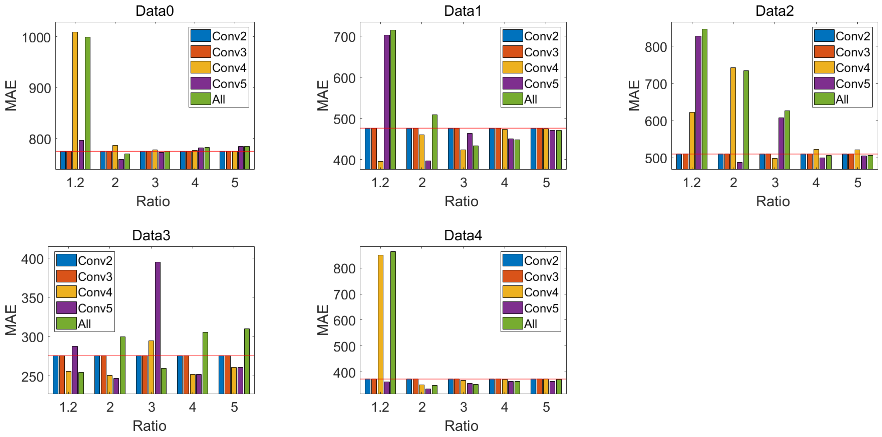

Figure 6: The MAE results of the estimations obtained on different subsets of the UCF dataset by pruning different layers with different ratios η.

and meanwhile reduce the size of the models.

Acknowledgments

This work was partly supported by National Natural Science Fund of China (No. 61725501), Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 20121102110032), and Rail Manufacturing CRC and Sydney Trains with UTS project ID PRO17-3968.

The first and the second authors have equal contributions to this paper.

References

References

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (7553) (2015) 436.

[2] C. Gao, P. Li, Y. Zhang, J. Liu, L. Wang, People counting based on head detection combining adaboost and cnn in crowded surveillance environment, Neurocomputing 208 (2016) 108–116.

[3] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, arXiv preprint arXiv:1804.02767.

[4] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Computer Vision (ICCV), 2017 IEEE International Conference on, IEEE, 2017, pp. 2980–2988.

[5] V. A. Sindagi, V. M. Patel, A survey of recent advances in cnn-based single image crowd counting and density estimation, Pattern Recognition Letters 107 (2018) 3–16.

[6] D. Onoro-Rubio, R. J. López-Sastre, Towards perspective-free object counting with deep learning, in: European Conference on Computer Vision, Springer, 2016, pp. 615–629.

[7] D. B. Sam, S. Surya, R. V. Babu, Switching convolutional neural network for crowd counting, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, 2017, p. 6.

[8] S. Kumagai, K. Hotta, T. Kurita, Mixture of counting cnns: Adaptive integration of cnns specialized to specific appearance for crowd counting, arXiv preprint arXiv:1703.09393.

[9] L. Zeng, X. Xu, B. Cai, S. Qiu, T. Zhang, Multi-scale convolutional neural networks for crowd counting, in: Image Processing (ICIP), 2017 IEEE International Conference on, IEEE, 2017, pp. 465–469.

[10] Y. Cheng, D. Wang, P. Zhou, T. Zhang, A survey of model compression and acceleration for deep neural networks, arXiv preprint arXiv:1710.09282.

[11] P. Molchanov, S. Tyree, T. Karras, T. Aila, J. Kautz, Pruning convolutional neural networks for resource efficient inference, arXiv preprint arXiv:1611.06440.

[12] H. Li, A. Kadav, I. Durdanovic, H. Samet, H. P. Graf, Pruning filters for efficient convnets, ICLR2016.

[13] S. Han, H. Mao, W. J. Dally, Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, ICLR2016.

[14] Y. Li, X. Zhang, D. Chen, Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 1091–1100.

[15] S. Srinivas, R. V. Babu, Data-free parameter pruning for deep neural networks, arXiv preprint arXiv:1507.06149.

[16] K. Ullrich, E. Meeds, M. Welling, Soft weight-sharing for neural network compression, arXiv preprint arXiv:1702.04008.

[17] M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European conference on computer vision, Springer, 2014, pp. 818–833.

[18] C. Arteta, V. Lempitsky, J. A. Noble, A. Zisserman, Interactive object counting, in: European Conference on Computer Vision, Springer, 2014, pp. 504–518.

[19] H. Idrees, I. Saleemi, C. Seibert, M. Shah, Multi-source multi-scale counting in extremely dense crowd images, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2013, pp. 2547–2554.

[20] A. B. Chan, Z.-S. J. Liang, N. Vasconcelos, Privacy preserving crowd monitoring: Counting people without people models or tracking, in: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE, 2008, pp. 1–7.

[21] Y. Zhang, D. Zhou, S. Chen, S. Gao, Y. Ma, Single-image crowd counting via multi-column convolutional neural network, in: Proceedings of the IEEE conference on computer vision and pattern recognition, IEEE, 2016, pp. 589–597.

[22] R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, R. López-Sastre, S. Maldonado-Bascón, D. Onoro-Rubio, Extremely overlapping vehicle counting, in: Iberian Conference on Pattern Recognition and Image Analysis, Springer, 2015, pp. 423–431.

[23] C. Shang, H. Ai, B. Bai, End-to-end crowd counting via joint learning local and global count, in: Image Processing (ICIP), 2016 IEEE International Conference on, IEEE, 2016, pp. 1215–1219.

[24] J.-H. Luo, J. Wu, W. Lin, Thinet: A filter level pruning method for deep neural network compression, ICCV.

[25] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531.