Visualization and Pruning of SSD with the base network VGG16

Xuemei Xie [†]*, Xiao Han, Quan Liao, Guangming Shi*
School of Electronic Engineering, Xidian University

xmxie@mail.xidian.edu.cn

ABSTRACT

This paper considers the state of art real-time detection network single-shot multi-box detector (SSD) for multi-targets detection. It is built on top of a base network VGG16 that ends with some convolution layers. Its base network VGG16, designed for 1000 categories in Imagenet dataset, is obviously over-parametered, when used for 21 categories classification in VOC dataset. In this paper, we visualize the base network VGG16 in SSD network by deconvolution method. We analyze the discriminative feature learned by last layer conv5_3 of VGG16 network due to its semantic property. Redundancy intra-channel can be seen in the form of deconvolution image. Accordingly, we propose a pruning method to obtain a compressed network with high accuracy. Experiments illustrate the efficiency of our method by comparing different fine-tune methods. A reduced SSD network is obtained with even higher mAP than the original one by 2 percent. When only 4% of the original kernels in conv5_3 is remained, mAP is still as high as that of the original network.

CCS Concepts

• **Computing methodologies→Machine learning→Machine learning approaches→ Neural networks.**

Keywords

Convolutional neuron network, Single-shot multi-box detector, Over-complete dictionary, Deconvolution, Pruning.

1. INTRODUCTION

Convolutional neural networks (CNNs) achieve remarkable performance in object detection problems [1-5]. The state of art detection network single-shot multi-box detector (SSD) [5] is considered in this paper since it is a real-time end-to-end CNN. SSD network, trained for 21 categories, constructs an over-complete dictionary with heavy redundancy due to its huge base network VGG16 [6] designed for 1000 categories. Because of the highly non-convex property of SSD. It is necessary to overcome the over-fitting problem of over-parameterization and the negative impact of random initialization.

Some attempts have been made to study and prune the redundancy of CNNs. [7] reduces the parameters in CNNs with low rank matrix factorization. [8] and [9] both use tensor low rank expansions technique to accelerate CNNs. However, they do not really reduce the redundant atoms in the dictionary constructed by a CNN, which may lead to over-fitting. There are also some works trying to optimize the model size of CNN from other perspectives, including sparsity regularization, connection pruning and low rank approximation. [10] applies regularization in network training, in order for a sparse structured CNN. [11] learns the importance of connections in CNNs, and remove the unimportant ones without decreasing accuracy. However, sparsity regularization and connection pruning approaches often produce non-structured random connectivity in CNNs. [12] proposes an algorithm for computing low rank tensor decomposition to remove the redundancy in the convolution kernels. [13] uses low rank representations of convolutional filters to create computationally efficient CNNs. [14] finds that the outputs of some neurons are always zero regardless of different inputs, so zero activation neurons are removed without affecting the overall accuracy. [15] prunes hidden nodes of a fully trained CNN according to an importance function and retunes the reshaped CNN using back-propagation. These methods can only evaluate the network performance by accuracy, while causal relationship between high accuracy and good feature leaning is somehow weak. In fact, a good CNN learns the similar features in hidden layers as the discriminative features of the dataset. It is efficient to evaluate model performance through visualizing the intermediate feature learned by a CNN, and the same goes for prunning the network redundancy.

In this paper, an iterative network pruning method is proposed by analyzing the redundant kernels in SSD network using visualization method. Intermediate features are visualized by deconvolution method [16] to find out the repeated kernels that have learned the similar feature. And we explore the active neuron [17] by comparing its activity over the test images. The inactive ones are pruned to reduce the redundant parameters and retain good initials for retraining the reduced model. We iteratively prune and retrain based on the last result. The proposed iterative pruning method compresses the SSD network while maintaining the accuracy of the system, compared to randomly initializing model fine-tuning method. Experiments illustrate the efficiency of our method, and a reduced SSD network is obtained with higher mAP of 76.31% than 74.6% of the original network by 2 percents. When there is only about 4% of the kernels in conv5_3 remained, mAP is still kept as high as that of the original network.

ICDLT '17, June 2–4, 2017, Chengdu, China.
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-5232-1/17/06…$15.00
http://dx.doi.org/10.1145/3094243.3094262

2. OVER-COMPLETE DICTIONARY

An over-complete dictionary should contain moderate amount of atoms enough to include discriminative features and meantime avoid over-fitting. SSD network constructs an over-complete dictionary with too much redundancy.

2.1 Over-complete dictionary constructed by CNNs

A CNN constructs an over-complete dictionary containing atoms of all kinds of latent concepts in the dataset. There are so many equivalent filters, namely atoms, in CNN, which are built by kernels from different layers in a nonlinear cascaded way. Along with the growing of CNN scale, the number of equivalent filters are increasing exponentially, because the higher layers need to consider all combinations of filters in the shallower layers.

With the redundancy of filters, CNN is equipped with the powerful representation ability. While the large size of parameters may result in over-fitting. So, too many or too few filters are not the best choice of a CNN, which may affect its performance. An optimal dictionary size is expected to achieve the best test accuracy of a CNN in specific problems, which means that a good network should have moderate amount of kernels. The curve in Fig. 1. Shows the relationship between the kernel number and test accuracy in a one-layer network trained on MNIST dataset, where the number of kernels being 10 is the best.

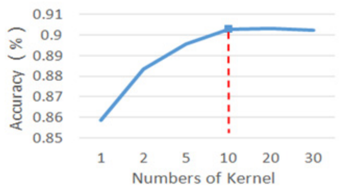


Figure 1. The relationship between the kernel number and test accuracy.

2.2 Redundancy in SSD network.

SSD network is a CNN, built on top of a base network VGG16 that ends up with some convolutional layers. It is a state of art single-shot multi-box object detector for multiple categories. The well-known original SSD network is trained on VOC2007 and VOC2012 dataset (16551 images), including 21 categories such as person and plotted plant. However, its base network VGG16, on trained on ILSVRC-2012 dataset (1.2M images), is designed for 1000 categories. Thus SSD network is over-parameterized for a small-scale problem of 21 categories.

SSD network constructs an over-complete dictionary with much redundancy, which may degrade the network performance. There is a great potential of compressing SSD model to obtain a simplified SSD network with fewer parameters but equivalent accuracy compared to the well-known original one.

Redundancy in SSD network exists mainly in 2 ways: inter-channel and intra-channel. Here we focus on the latter one, which is often taken in consideration for network pruning. The intra-channel redundancy is analyzed through deconvolution and a network pruning method is proposed, accordingly.

3. VISUALIZING REDUNDANT KERNELS OF VGG16 IN SSD

In this section, we aim to understand the intra-channel parameters redundancy in SSD intuitively via visualizing the feature representation of its base network VGG16, where conv5_3 layer is selected because it is the last layer for internal feature learning in VGG16.

3.1 Repeated kernels

Though variety of atoms in an over-complete dictionary can achieve effective feature extraction and representation, it may also involve much redundancy. The SSD network, based on a general network VGG16, has so many kernels in a single layer. With this over-parameterized structure, it constructs an over-complete dictionary with much redundancy. Unfortunately, this may result in some kernels learning the similar feature, which can be seen as repeated and thus can be pruned to compress the network.

Here we take conv5_3 layer as an example to show what the repeated kernels learn in SSD network. When the input image, shown in Fig. 2(a), is tested in SSD network, the detection result is human and plotted plant as shown in Fig. 2(b). By deconvolution method, we can visualize internal representations learned by different layers in a deep neural network. For the example image (see Fig. 2(a)), we visualize the features extracted by 147th, 208th and 447th kernels in conv5_3 layer, as shown in Fig. 3 (a), (b) and (c), respectively. It is obvious that these three kernels learn the similar feature "human head". The same goes for 32th, 37th and 270th kernels which extract the feature "plotted plant" shown in Fig. 3 (d), (e) and (f). Atoms, corresponding to these kernels, overlap so much, thus only one of them is enough to achieve the same feature extraction, which proves much redundancy in SSD structure. With the purpose of obtaining a proper network, we expect that different kernels extract different features separately by eliminating the redundancy, while composing the same representation of the input image, such as 447th, 160th, 270th kernels shown in Fig. 4.

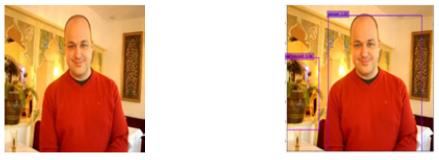


Figure 2. A test image of SSD and detection result with two objects. (a) Input image and (b) Detection result.

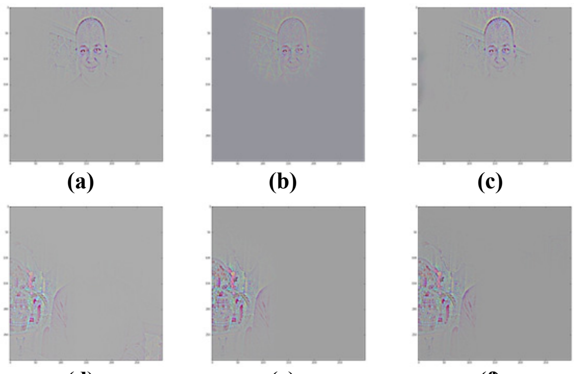


Figure 3. Features learned by repeated kernels in conv5_3. Feature "human head" extracted by (a) 147th, (b) 208th and (c) 447th kernels, "plotted plant" by (d) 32th, (e) 37th and (f) 270th kernels.

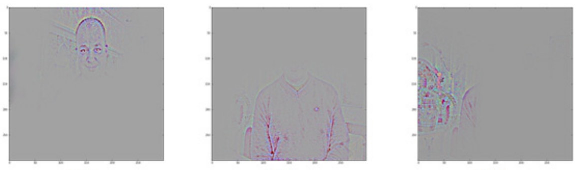
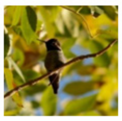
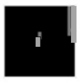
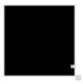
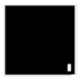


Figure 4. Representation of the input image held by different kernels. (a) 447th, (b) 160th, and (c) 270th kernels.

3.2 Inactive kernels

Apart from the repeated kernels, there are many kernels that are inactive in SSD, which means that they are hardly activated. To prove this point, we test 4618 images containing 21 categories, recording the numbers of each neuron in conv5_3 activated. Many kernels are activated few times throughout the testing process. Details can be seen in Tab. 1, taking the 83th kernel for example. By visualization of its feature maps in conv5_3 of several test images, it can be seen that the activations of this kernel are quite small compared to other active kernels, the top1 activations shown under the feature maps. This reveals that many kernels in the SSD structure do not learn effective features for detection task, and may degrade the performance instead.

Table 1. Activations of inactive kernels and active kernels.

Input image	Index=83	Index=321	Index=343
	2.02	16.26	24.95
	0.00	27.56	24.47
	0.80	10.11	32.15
	0.89	21.74	 23.96
	2.04	 38.63	 58.66

4. THE PROPOSED PRUNING METHOD

We compress the base network model by reducing the redundant atoms (kernels) in the consideration of precision and speed. A data-driven approach is proposed to estimate kernels in each layer. For a given layer, conv5_3 for an example, we present a network pruning method. It includes 3 steps,

- Sorting kernels by activity.
- Choosing active kernels as initials and fine-tuning (training) the network.
- Testing and giving the number of remaining kernels and the corresponding mAP.

In Step One, kernels in conv5_3 are sorted by activity. Here, in this paper, we test certain images in modeling the shortened VGG network of SSD. Those test images are randomly chosen from VOC2007 test dataset which has 9375 images in total. In the first time, 4618 images are set for the test and they are 512 kernels corresponding to 512 feature map. We chose the top1 of the value in each feature map. Then, for a single test image, there are total 512 top1 values. Among them, the top K values are left with their corresponding feature maps. Here, we set K=20 as an example. For 4618 images, there are total 4618*20 feature maps. Most of the are repeated. The number of the repetition of feature maps are sorted in descent order, with the index of kernel, as given Tab. 2. According to the rank of the number of the repeated feature maps, we choose certain number of the maximum values and take their corresponding kernels as active ones.

Table 2. Activation of the related kernels in conv5-3 of the original model.

4618 images over 9375 images are chosen randomly to test the active level of original SSD with **512** kernels

Index	Kernel Index	Number of repetition
1	343	3108
2	440	2931
3	249	2885
4	319	2323
⋮	⋮	⋮
508	149	1
509	131	1
510	136	1
511	369	1
512	83	1

Table 3. Activation of the related kernels in conv5-3 of the reduced model.

1535 images over 9375 images are chosen randomly to test the active level of original SSD with **100** kernels

Index	Kernel Index	Number of repetition
1	91	1375
2	42	1279
3	63	1207
4	41	1093
⋮	⋮	⋮
96	75	79
97	77	59
98	18	55
99	87	47
100	76	28

In Step Two, according to the repetition number in Tab. 2, the last 512-J (e.g. J=100) inactive kernels are pruned since they contribute little to detection task. While the first J ones are remained. Thus a reduced model with J kernels is obtained. The remaining kernel weights are initials for a reduced model training process. Since good initial guess accelerates convergence speed and keeps high accuracy for a nonlinear optimization problem, the reduced model is well trained.

In Step Three, we repeat the first two steps, the operation is performed on the base of reduced model in the second step, together with the testing results of the reduced model for its mAP. Tab. 3 shows the repetition number ranking of 368 kernels in the reduced model. The J kernels are chosen as the most active ones to construct the conv5_3 layer of the further reduced model, with the smaller number of kernels left.

Comparing with the random initial in fine-tuning, the proposed network pruning method takes an iterative optimization strategy, bringing good initials in each optimization process.

5. EXPERIMENTS

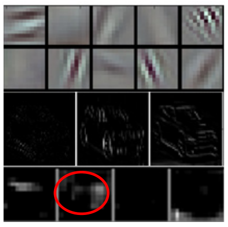
In this section, AlexNet, as an example, is trained on different initials to illustrate the positive effect of a good initial guess. The proposed network pruning method is used to obtain reduced SSD network. Since the proposed method retains active kernels as good initials, the reduced network has higher mAP of 76.22% than that of the original one.

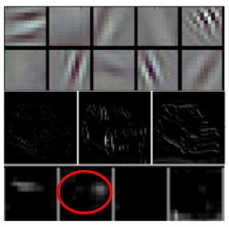
5.1 CNN fine-tuning on good initial

Initial guess has an effect on training CNNs, due to the highly nonlinearity property of the system. Here we train AlexNet, a classification network for 1000 categories, on different initials. It can be seen that the fine-tuned AlexNet has good filters and its feature map is sparse.

Input image shown in Fig. 5(a) is tested through AlexNet. Fig. 5(b) shows the kernel from covn1 (the first line) and the feature maps from conv1 and conv5 (the second and the third line), of the original network trained for 1000 categories. We mainly consider the following two methods to train AlexNet for classifying 2 categories of car and motor: (1) fine-tuning a network from the original, namely all the weighs are initialized by the original except fc8. (2) all the weights are initialized randomly. Kernels and feature map by 2 methods are shown in Fig. 5(c) and (d) respectively. It can be revealed from Fig. 5(b) and (c) that after fine-tuning, the kernels of conv1 have no obvious change. While for the feature map of conv5, it becomes sparse, especially in red circle. That is to say, for car and motor, the network, which is obtained by method (1) has better ability of feature representation compared to the original one. Comparing Fig. 5(c) and (d), random initialization is worse, since only kernels of the network obtained by method (1) have structural characteristics. We can conclude that good initialization of weights plays an important role on positively affecting the performance of a network.

(a)


(b)


(c)

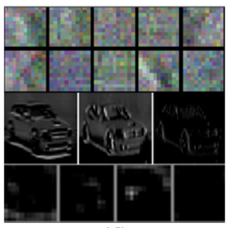

(d)

Figure 5. A test image and its feature maps from different networks with kernels in conv1 respectively. (a) Input image. (b), (c) and (d) kernels in conv1 (first line) and feature maps from conv1 and conv5 (second line and third line).

5.2 SSD network pruning by the proposed method

Due to the effect of initials, a network pruning method is proposed in section 4 to retain good kernels, namely good initials, by iteratively fine-tuning CNNs.

SSD network is pruned by 3 methods and 3 mAP curves are shown in Fig. 6: the red curve corresponds to the proposed method, the green one to the direct method and grey one to the random method. Direct method includes 2 steps similar to the first 2 steps of the proposed method without iteration. Random method prunes SSD network by random initials.

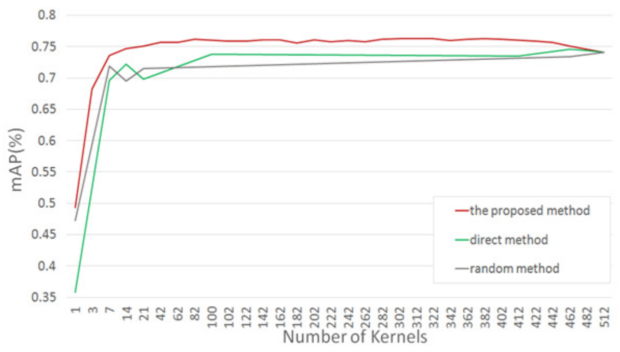


Figure 6. mAP curves of the reduced SSD networks using 3 pruning methods, red one to the proposed method, green one to the direct method and grey one to the random method.

It can be concluded that:

(a) The red curve achieves highest mAP compared to the others, since the proposed method retains good kernels in each iteration.

(b) The reduce SSD network, by the proposed method, has higher mAP of 76.31% than the original one in [5] with mAP of 74.30% by 2 percent, since redundancy in the original model is removed to avoid over-fitting.

(c) When the number of kernels is 322, a peak value of 76.31% emerges in the red curve, meaning that the size of the over-complete dictionary is optimal. Moreover, when there are only 21 kernels, namely about 4% of the original, remained, mAP is still as high as 74.67%. The SSD network can be reduced much within a reasonable error rang.

An input image in Fig. 2(a) is tested through several reduced networks with different numbers of kernels. Detection results are shown in the first column in Fig. 7, with the confidence of "potted flower" (conf$_1$) and confidence of "person" (conf$_2$). In the second column are feature maps from different reduced models with their corresponding numbers of kernel in conv5_3. Deconvolution results of the “potted flower” and “person”, in the third and fourth column, are obtained by projecting the maximum coefficient in a neuron to the image space for visualizing how the “potted flower” and “person” activates the kernels.

It is obvious that when the number of kernel=21, the detection results are better than that of the 512, since less kernels with good initials can be more focused on the discriminative features in training process. Comparing the second column, feature maps are more active as number of kernel decreases.

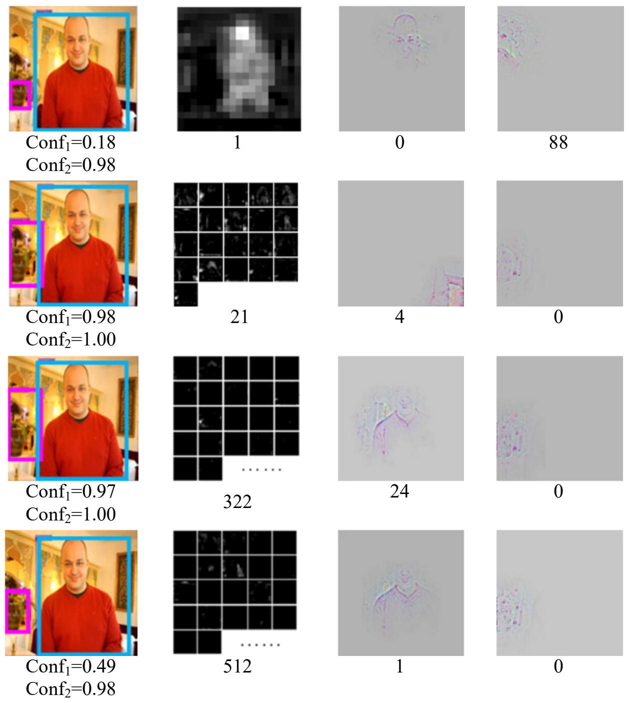


Figure 7. Detection results in the first column with confidence of “potted flower” and “person” corresponding to conf1 and conf2, feature maps in the second column with the number of kernel in conv5_3 and deconvolution results of “potted flower” and “person” in the third column with the coefficient indexes.

6. CONCLUSION

A CNN constructs an over-complete dictionary and it should be in appropriate size. Due to a large base network VGG16, there are many redundant dictionary atoms in SSD network. Visualizing features learned by SSD network can be guidance to evaluating network performance and pruning redundant kernels. Through analyzing the redundancy by deconvolution and data-driven methods, an iterative network pruning method is proposed to remove inactive kernels of conv5_3 layer in this paper. Since over-fitting is avoided, the reduced SSD network achieves better performance with fewer parameter, less storage and less calculation time than the original one. Experiments illustrate the efficiency of our network pruning method.

7. ACKNOWLEDGMENTS

This work is supported in part by Natural Science Foundation (NSF) of China (61472301).

8. REFERENCES

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, pp. 91-99, Jun. 2016.

[2] R. Girshick, "Fast R-CNN," in *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, pp. 91-99, Jun. 2016.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, *et al.*, "SSD: Single-shot Multibox Detector," in *European Conference on Computer Vision (ECCV)*, 2016.

[6] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-scale Image Recognition," *arXiv preprint arXiv*:1409.1556, 2014.

[7] M. Denil, B. Shakibi, L. Dinh, N. de Freitas, et al. “Predicting parameters in deep learning,” in *Advances in Neural Information Processing Systems*, 2013.

[8] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman, "Speeding up Convolutional Neural Networks with Low-rank Expansions, "*arXiv preprint arXiv*:1405.3866, 2014.

[9] E. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting Linear Structure within Convolutional Networks for Efficient Evaluation, " in *Advances in Neural Information Processing Systems*, 2014.

[10] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky, "Sparse Convolutional Neural Networks, " in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[11] Song Han, Jeff Pool, John Tran, and William Dally, "Learning Both Weights and Connections for Efficient Neural Network," in *Advances in Neural Information Processing Systems*, 2015.

[12] Yani Ioannou, Duncan P. Robertson, Jamie Shotton, Roberto Cipolla, and Antonio Criminisi, "Training CNNs with Low-rank Filters for Efficient Image Classification," *arXiv preprint arXiv*:1511.06744, 2015.

[13] Cheng Tai, Tong Xiao, Xiaogang Wang, and Weinan E, "Convolutional Neural Networks with Low-rank Regularization," *arXiv preprint arXiv*:1511.06067, 2015.

[14] Hu, Hengyuan, et al. "Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures," *arXiv preprint arXiv*:1607.03250, 2016.

[15] He, Tianxing, et al. "Reshaping Deep Neural Network for Fast Decoding by Node-pruning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014.*

[16] Zeiler, M. D. and Fergus, R, "Visualizing and Understanding Convolutional Networks," in *Europeon Conference on Computer Vision (ECCV),* 2014.

[17] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object Detectors Emerge in Deep Scene CNNs," in *International Conference on Learning Representations (ICLR)*,2015.